DU
JEAN
MONTEREY

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

G557

FEASIBILITY STUDY OF A MICROPROCESSOR
CONTROLLED ACTUATOR TEST MECHANISM

by

Gregory Lawrence Goode

March 1988

Thesis Advisor: Robert H. Nunn

Approved for public release; distribution unlimited

T238931

REPORT DOCUMENTATION PAGE

REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited
DECLASSIFICATION / DOWNGRADING SCHEDULE	
ERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) 69Nn	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
NAME OF FUNDING / SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

TITLE (Include Security Classification)
ASIBILITY STUDY OF A MICROPROCESSOR CONTROLLED ACTUATOR TEST MECHANISM.

PERSONAL AUTHOR(S)
Goode, Gregory L.

TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM _____ TO _____	1988 March	102

SUPPLEMENTARY NOTATION
he views expressed in this thesis are those of the author and do not reflect the official
licy or position of the Department of Defense or the U. S. Government."

COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Microprocessor, Hydraulics, Digital control.

ABSTRACT (Continue on reverse if necessary and identify by block number)
is thesis describes the investigation of the feasibility of using a commercially available
crocomputer to control and test a missile fin actuator. Topics discussed include system
delling, automated data acquisition, system identification, simulation and controller
sign. Modularity, both functional and conceptual, is stressed in the design process as well
integration of modules during the modelling and simulation process. Verification of the
mputer simulation is used extensively as an interactive tool to modify the system model.
e hybrid system under investigation contains analog and discrete components some of which
both nonlinear and discontinuous. The use of digital systems, their limitations and
vantages are highlighted in the modelling of these components and the development of a
ntrol system.

DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
NAME OF RESPONSIBLE INDIVIDUAL bert H. Nunn	22b TELEPHONE (Include Area Code) 408-646-2365 22c OFFICE SYMBOL 69Nn

FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

☆ U.S. Government Printing Office: 1986—606-243

Approved for public release; distribution is unlimited

Feasibility Study of a Microprocessor Controlled Actuator Test Mechanism

by

Gregory L. Goode
Lieutenant, United States Navy
BSEE, University of Texas, 1979

Submitted in partial fulfillment of the
requirement for the degrees of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

and

MECHANICAL ENGINEER

from the

THESIS DISCLAIMER

The suitability of either the computer software generated during this investigation or the commercial programs discussed in this thesis for a particular application is unwarranted. While considerable effort has been expended to insure reliable operation in the context of this research effort, logical or computational errors may exist in the programs. Use of any and all software derived from this thesis by the reader is at his own risk.

ABSTRACT

This thesis describes the investigation of the feasibility of using a commercially available microcomputer to control and test a missile fin actuator. Topics discussed include system modelling, automated data acquisition, system identification, simulation and controller design. Modularity, both functional and conceptual, is stressed in the design process as well as integration of modules during the modelling and simulation process. Verification of the computer simulation is used extensively as an interactive tool to modify the system model. The hybrid system under investigation contains analog and discrete components some of which are both non-linear and discontinuous. The use of digital systems, their limitations and advantages are highlighted in the modelling of these components and the development of a control system.

TABLE OF CONTENTS

List of Figures

ACKNOWLEDGEMENTS

The author would like to take this opportunity to express his appreciation for the patience, guidance and humor demonstrated by Professors Robert H. Nunn and D. L. Smith in support of this thesis. The author also thanks Professor A. J. Healey for his cheerful late night discussions and Professor Shankar Lal for his example of boundless enthusiasm in the pursuit of knowledge.

I. INTRODUCTION

A. BACKGROUND

The Naval Postgraduate School in association with the Weapons Powers Systems Branch, Code 3275, of the Naval Weapons Center (China Lake), has undertaken an investigation of the feasibility of the simulation and testing of missile fin actuators with a microcomputer based system. In the past, each time a new fin actuator has been proposed, a unique testing system has been designed to determine the basic response parameters of the actuator. A programmable data acquisition and test system has the potential of providing a more comprehensive test environment while reducing the time, resources and cost required to evaluate and predict actuator performance. Previous attempts to utilize general purpose microcomputers in this application have failed to achieve the required sample frequencies or precision necessary for real time control.

This particular segment of the investigation concerns itself with the simulation and load testing of control surface actuators. The top level requirements (TLR) of a particular missile design impact the formulation of the apparatus inasmuch as the missile flight profile and envelope determine the required torque, power and precision of the actuator. An apparatus designed to simulate and test

a variety of actuators must possess the capability and flexibility of providing the various loads and measuring the response of a candidate actuator. Furthermore, the system software should include the ability to acquire, store and analyze the test data to allow modification of the simulation model.

1. Objectives

It is the objective of this investigation to demonstrate the feasibility of using a general purpose microcomputer to direct and control a missile fin test apparatus. In support of this objective, the system must achieve the following goals: 1) accommodate a wide range of fin actuators, 2) allow modification of control methods through software changes, 3) store and retrieve flight profiles and actuator response data, 4) provide a means of predicting the actuator performance beyond the physical bounds of the test apparatus, and 5) provide software capable of analyzing actuator response to diagnostic maneuvers. This study directly supports advances in missile powering and control technologies.

2. Advantages of Microprocessor Control

Programmable control is by its very nature flexible. Control algorithms are easily modified or replaced with improved versions. Non-linear and discontinuous functions

difficult or impossible to implement on analog devices are readily employed. The accuracy and repeatability of the controller is unaffected by aging and constant realignment is not required. Additionally the microprocessor can be utilized as a processor both preceding and following control action. Microprocessor control allows the modification of the controller logic subject to changes in the operating environment or the plant parameters (adaptive control).

B. SYSTEM DESCRIPTION

The proposed physical system is characterized by a high degree of modularity to allow for replacement of component parts. Three functional modules (Fig. 1.) make up the system. The actuator, its gear trains and linkages, power supply and control systems (if required) comprise the actuator module. The actuator, whether pneumatic, electrical, hydraulic or a combination of these, presents unique modeling, control and interfacing challenges. However, each actuator is required to position the fin within prescribed tolerances and time limits (TLR). The input and output variable of interest is fin position as a function of time.

The second module simulates the load seen by the actuator during a flight profile. The load module mimics the required torque based on fin aerodynamic characteristics, angle of attack and missile flight dynamics. Aerodynamic forces, drive train losses and hysteresis effects are accounted for in this module. The input and output variable is torque as a function of time.

The third module contains the logic, programs and hardware for command and control of the actuator and load modules. This module by necessity must be programmable to implement various control algorithms unique to each actuator and load device. Fully integrated systems could utilize multiple channel, continuous or state space control configurations as appropriate.

The transducers, sensors and interfaces required to integrate the modules into a functional system are grouped within the appropriate module.

The selection of the components for this study has been strongly influenced by the available equipment on hand. The particular components used do not necessarily reflect the investigator's opinion as to optimal choices for final system design.

1. Hardware

a. Control Module

The control module (Fig. 2.) consists of a Zenith Z-248 microcomputer operating at 8 MHz, a Data Translation (DT) high speed data acquisition card (DT2821) and the software generated to operate the system. Library routines (AT-LAB, NOTEBOOK) are available to support a wide variety of data acquisition needs including 16-channel analog to digital conversion and 2-channel digital to analog conversion. All software, produced as a result of this study, is currently written in MicroSoft pascal and compiled to executable machine code.

b. Actuator Module

The actuator module (Fig. 3.) selected for evaluation in this study consists of a Helac planetary hydraulic rotary actuator, a Vickers model SM4-15 servovalve and a Vickers EM-D-20 servo amplifier. Hydraulic fluid is supplied at a maximum pressure of 1000 PSIG. Servo amplifier gain has been adjusted to allow full stroke of the servovalve to correspond to a ±10 volt DC input. Actuator position is sensed by a direct-mounted rotary variable differential transformer (RVDT) that provides analog voltage output.

c. Load Module

The load module (Fig. 4.) selected for evaluation in this study consists of a Magtrol hysteresis absorption dynamometer, model 4637 torque controller, modified to accept analog voltage input and a model 4618 dynamometer readout. The readout provides both speed and torque analog output as well as a visual digital display. In the present configuration only torque data is utilized for control purposes. This investigation has shown that a serious limitation is imposed by the use of a dynamometer as a loading device, because such a device can only apply a torque in opposition to the rotation of the actuator. This presents a limitation in the system if the static torque of the actuator module is less than the torque applied by the aerodynamic forces on the fin.

d. Interfacing

All signal and data channels between modules are expected to be less than or equal to ±10 volt DC. Linear amplifiers are utilized to adjust data and signal levels. Signal conditioning whether it be analog to digital (A/D), digital to analog (D/A), voltage to current, etc. is expected to take place within the appropriate module. Required conversions are treated as a delay (if appreciable) during simulation.

2. Software

a. Programming Language

Control programs are written in MicroSoft pascal [Ref. 1.] and compiled to run under either the PC DOS or MS DOS operating systems. Pascal's ability to manipulate text and data played a large part in its selection over other languages such as FORTRAN or C. Of special interest was the feature of true recursive procedural calls available in pascal [Ref. 2.], and the ability to access FORTRAN subroutines.

Cycle time, the time required to first acquire then output a value to an external device is critical to the overall loop time of a digital control system. The cycle time is often larger than the period required for calculation and evaluation of the control variables. Languages that store data in sequential files (BASIC) are considerably slower than those that utilize random access files (pascal, FORTRAN) and were avoided.

b. Simulation Tools

Simulation of the component, subsystem and system models was accomplished with a software package written by Integrated Systems, Inc. : MATRIXx. This software allows matrix calculations with graphical output.

It is designed for use by control engineers in applications dealing with classical and state space control design.

SYSTEM_BUILD, a subsection of MATRIXx, provides a graphical environment for building, modifying and executing simulations. The program is interactive and menu-driven and allows on-screen construction of continuous-time, discrete-time, linear, non-linear or multi-rate simulations. In addition to a standard library of basic building blocks the user can integrate special purpose FORTRAN routines of his own design into the simulation. One-step and multi-step integration methods are provided as well as stiff system solvers.

SYSTEM_ID, another subsection of MATRIXx, addresses data analysis, parametric and non-parametric system identification and digital filtering.

c. Programs

Numerous programs were developed to align, exercise and test the system. These programs were written in pascal and take advantage of the structured format of the language. Unlike FORTRAN a procedure (subroutine) may recursively call itself. The programs and flowcharts are listed in Appendix B. Each program contains self documenting comments enclosed within braces ({}). A brief description of three programs follows:

* POSIT.PAS

The simplest of the programs, POSIT demonstrates implementation of a negative feedback loop for the positioning of the actuator. Either proportional or proportional plus integral control may be selected by simple modification of the program. The user is queried for desired position and gain, a check performed to insure range validity, the actuator positioned and the user again prompted for input values. The program is terminated by selecting a position of zero degrees.

* CYCLE.PAS

Similar in structure to POSIT, CYCLE exercises the system by cycling the actuator to insure thermal equilibrium throughout the servomechanism prior to test runs. Elements of POSIT are used as procedures contained within CYCLE.

* PROFILE.PAS

PROFILE demonstrates the systems ability to both control the actuator in real time and act as a data acquisition system. The user may create a load and position schedule or select a standard (previously defined) schedule for test. The program executes the schedule and stores desired and actual positions and loads. Each element in the schedule array is considered a step input to the system. The test program sequences through the array in discrete time steps (user selected) and attempts to attain desired

fin position and torque loads. Modification of the timing loop (outer) allows for time intervals as small as one hundredth of a second. The torque and position control loops (inner) are presently configured as simple feedback loops. The control algorithm continues to execute (free running) during the "present" time interval. Loop time (inner) achieved with this configuration (with computation), is on the order of ten milli-seconds.

3. Data Acquisition System

Digital based data acquisition systems simplify the collection and processing of data. Analog signals manifest infinite resolution, but representation by analog devices (sensors) rarely allows for a precision of more than three or four decimal digits. Further, the accuracy of the analog sensor is directly related to the amplitude, or range, of the signal to be measured.

Theoretically the precision of a digital sensor can be extended indefinitely, but the utility of such an exercise is questionable in that increased precision requires an increased expenditure of time. Resolution of twelve or sixteen binary bits is sufficient for most engineering applications. Digital data is inherently noise-resistant [Ref. 3. p. 175]. Accuracy of the data obtained is based on the sampling frequency and resolution and not the amplitude of the signal.

a. Hardware

Data acquisition and control signals are directed through a Data Translation high speed interfacing board (DT 2821). The board supports sixteen digital input/output (I/O) channels, sixteen twelve bit analog to digital (A/D) input channels and two twelve bit digital to analog (D/A) output channels. The analog I/O channels may be configured unipolar or bipolar, single ended or differential. Throughput, one complete read or write cycle, is limited to fifty kilohertz.

b. Software

The low level library routines contained in the ATLab package [Ref. 4.] provide the user with a wide array of procedures, functions and subroutines in FORTRAN, pascal and C. These library routines include single and multiple-channel A/D and D/A conversion as well as buffered file storage and retrieval. A high level software package (Lab Notebook [Ref. 5.]) that supports the DT2821 has been used extensively for data acquisition during the frequency response testing of the modules. This package offers the user the convenience of menu driven procedures but is slower and less flexible than direct manipulation of the DT 2821.

C. APPROACH

In keeping with the objectives of the investigation, the overall approach strives to compartmentalize the modelling and analysis of the system under investigation. Rosko [Ref. 6.] describes the simulation design process as a multistep iterative procedure consisting of the following steps: 1) Isolate the system and define inputs and outputs, 2) determine a mathematical model that accurately describes the system, 3) develop algorithms that implement the approximate discrete equivalent model of the continuous system, and 4) test the simulation to modify steps 1 thru 3. The iterative procedure continues until formal acceptance criteria are satisfied or until resources (time, funds, etc.) are exhausted.

1. Control Systems Specifications

Specifications for a system may be classified into the following five groups:

* Steady State Accuracy
* Transient Accuracy
* Noise Rejection
* Parameter Sensitivity
* Control Effort

Generally, the system must satisfy criteria taken from all five categories.

A "Type 0" continuous system exhibits a finite nonzero steady-state error to a constant input (zero order polynomial input). "Type 1" continuous systems demonstrate a finite steady state error in response to a ramp input (first order polynomial). Discrete control systems, due to the limitations inherent in sampling and time delay, display both a finite transient and steady state error. This apparent disadvantage is somewhat offset by the relative insensitivity of the discrete system to high frequency, low amplitude noise. No general comparison between continuous and discrete controllers for parameter sensitivity or control effort is advisable without regard to a specific application.

2. Stability

In simple terms, stability can be thought of as the bounded response of a system to bounded stimuli. Åström and Wittenmark [Ref. 7. p. 94] define stability in terms of

solutions to state space equations. Given a state space model defined by

$$x(k + 1) = \Phi\ x(k) + \Gamma\ u(k) \qquad (1)$$

and

$$y(k) = \Omega\ x(k) \qquad (2)$$

where :

k = the present state

x = state vector

u = input vector

y = output vector

Φ = state coefficient matrix

Γ = input coefficient matrix

Ω = output coefficient matrix

the stability of the system may be directly determined by calculation of the eigenvalues of the matrix Φ. If the matrix has parameters as coefficients this method cannot be used. Calculation of the characteristic polynomial and determination of its zeros may be indicated as well as application of the root locus method or the Nyquist criterion. However for non-linear systems Lyapunov's Second Method [Ref. 12 p. 721] is generally preferred if a Lyapunov function can be determined.

Simulation can be useful in determining the behavior of a system and is indicated if the system exhibits

pathological non-linearities such as discontinuities. However, the combination of analysis and simulation will generally prove superior to either method used alone.

The system under investigation has no inherent restoring force, and as such, no equilibrium states. Stated another way, the system (neglecting frictional effects) exhibits no lowest energy configuration. The system is metastable when energized. To prevent limit cycle behavior, a dead band (tolerance) must be introduced into the controller either during the computation of the error signal or inherently by the precision of the samplers to prevent cyclic control action about the desired position of the actuator.

II. MODELLING AND DESIGN

A. METHODS

Each component is modelled, either by deriving equations that represent the physics of the component (first principles), or by approximating experimental test data. If the proposed models exhibit linear behavior, analysis of the system proceeds along classical lines.

Transform techniques such as the methods derived by Nyquist (1932), Bode (1945), Wiener (1948) and Evans (1950) deal with classical control design problems (linear time invariant continuous systems). Each make use of either Laplace or Fourier transform representations of dynamic systems to analyze the performance of a system. The analysis of linear time invariant discrete systems requires additional transformations but with computer aided design (CAD) packages, and interactive graphics, the iterative application of these techniques is greatly simplified.

Hybrid systems such as the system under investigation, defy classical analysis. The normal approach relies on the ability of the investigator to linearize the system about an operating point and infer system behavior from solutions generated from the linearization. Analytic solutions at selected points within the range of expected operation of

the system are then integrated to provide continuous coverage. Systems with multiple or discontinuous non-linear attributes require simulation to predict operating parameters. If the system can be linearized, a state space model is constructed and analysis proceeds along well-established lines (transient response, frequency analysis, etc.). If linearization is inappropriate, only a full system simulation is adequate to predict performance over a wide range of operating points.

1. First Principles

Modelling by the use of first principles is indicated when the underlying dynamic characteristics of the components and system under investigation are well understood. Most continuous physical systems such as electric circuits and masses subject to accelerations can be modelled by differential equations. If the components are linear and time invariant their treatment is straightforward. Even in the event the system is time varying it may be advantageous to formulate an idealized linear time invariant model. On the other hand if the underlying principles are not well understood or if the system is highly non-linear it may be more efficient to utilize experimental means to determine an acceptable model.

2. Experimental Analysis

The determination of an analytical system model may be difficult. Experimental analysis often provides an acceptable alternative to conventional analysis. Of the available techniques, frequency response methods are generally considered the most useful in determining the approximate system transfer function. Asymptotic approximation of the Bode plots (amplitude, phase) and judicious manipulation of the corner frequencies can provide the investigator with suitable approximations of the system behavior. In the event the system is highly non-linear, exhibiting such characteristics as saturation, deadband and hysteresis, the investigator must consider the effects of input amplitude. In addition to sinusoidal inputs the analysis of the system step response is often illuminating. Least square estimation and maximum likelihood estimation of system parameters are described by Borrie [Ref. 8. p. 273-285] as the two established methods for dealing with system identification. Both methods rely on elaborate computer analysis routines. Least square methods are generally most useful when dealing with "white" noise, and require a large data base for analysis. Maximum likelihood methods assume some knowledge of the character of the noise terms, and rely on a "weighted" augmented matrix.

B. CONTROL MODULE

Considerable flexibility is inherent in the selection of a programmable control module. To modify the method of control requires only that the control program be modified, not the system hardware. Implementation of basic control actions (proportional, integral, differential) is straight forward for single input single output (SISO) systems and easily adapted to multiple input multiple output (MIMO) systems. Additionally, time varying control systems are much simpler to design and configure within the scope of a control program than in implementation of physical hardware.

1. Discrete Systems

Unlike continuous systems, discrete control systems depend upon sampled data. An ideal sampler reproduces the value of a signal at a specified time (sample time). The sample value can be considered an impulse with a magnitude equal to the signal value. This process can be represented by a impulse modulator [Ref. 8. p. 73]. In a microcomputer controlled system, this sample value is processed by the computer program. The output signal (quantized) is either maintained constant during computation (zero order hold) or

modified by a polynomial function (data extrapolation) [Ref. 10. p. 82]. The transfer function of a zero order hold in continuous space is defined by

$$G(s) = (1 - \epsilon^{-Ts})/s \qquad (3)$$

where ϵ is arbitrarily small and T represents the time between samples. In effect the zero order hold acts as a low pass filter. A sine wave and its sampled representation are shown in Fig. 5.

Digital systems include a further complication. The amplitude of the signal is quantized. The sampled value resolution is limited by the number of bits selected to represent the numerical value. In theory this value can be made as small as one desires. However in practice accuracy beyond three or four decimal digits (which corresponds to approximately twelve binary bits) is sufficient for most control applications. Sampling and quantizing make up the analog to digital (A/D) conversion process. The inverse operation of digital to analog (D/A) conversion takes place when a stream of impulse modulated values are acted upon by a low pass filter. Of immediate importance is the phase lag introduced. In feedback control systems the phase shift is a destabilizing effect of the conversion. The severity of the destabilizing effect depends on the sampling frequency and the lowpass filter used.

Shannon's Sampling Theorem states that any signal with highest frequency component f_h, can be uniquely determined by the values of any set of sampled events collected at a frequency of twice f_h. Åström and Wittenmark [Ref. 7. p. 31] recommend that reasonable sampling rates for control purposes reside within a range of six to ten times the bandwidth of the signal, or two to three times the signal rise time. Sampling at frequencies below twice f_h allows aliasing or frequency folding [Ref. 11. p. 170]. This results in the sampled signal exhibiting frequencies much different than those found in the original signal. Such frequencies are termed alias frequencies.

From a design standpoint, the prudent engineer must insure the selection of both a sampling frequency well above the highest frequency of interest, and that the quantization error inherent in the selection of the A/D converter remains negligible. If frequencies exist above the sampling frequency, analog filtering of these components is indicated prior to sampling.

2. Control Design

Linear systems and methods for their control are well established in the literature. Non-linear and time variant systems generally receive describing function or phase-plane analysis [Ref. 12. p. 531-620]. The actuator model under investigation has no "restoring force"

associated with it and without control action, a disturbing force results in an ever increasing response. With control action, the phase plane trajectory encounters a region defining a stable limit cycle as it approaches the origin.

Although the system to be controlled in this study exhibited non-linear behavior, a simple easily described control algorithm had the requisite reliability to control the plant, and was sufficient for test and evaluation at this level. The inherent limit cycle behavior was circumvented by assigning a error tolerance within the control program.

Proportional control was selected as the control method with an option for proportional plus integral control. Two saturation conditions were imposed on the control system model by the physical and computational limitations of the module. First, the control module I/O board was limited to a voltage output of ±10 volts DC. Second, the signal error variable (type integer) in the control program was limited in magnitude by the program language compiler. The control system model is summarized in Fig. 6.

C. ACTUATOR MODULE

The solid state analog amplifier (Fig. 7.) that provides current to the control valve torque motor windings saturates at 250 milliamps and exhibits a break point (-3

dB) roll off at 1800 rad/sec. The amplifier performance was determined by frequency response testing [Ref. 12. p. 451].

The control valve (Fig. 8.) can be considered a two-stage critical-centered spool valve for the purpose of analysis. The position of the spool, load, supply pressure, valve pressure drop, compressibility of the fluid and compliance of the fluid containment all play a part in determining the flow of hydraulic fluid to the actuator ram. Each land and seat acts as a variable orifice that meters flow through the valve. Flows through variable orifices are described by the simplified orifice equation [Ref. 13. p. 359-370], as follows :

$$Q = C_d \, A \, [2/\rho \ \Delta P]^{\frac{1}{2}} \qquad (4)$$

Q = mass flow rate

C_d = discharge coefficient

A = area

ρ = mass density

ΔP = pressure difference

An approach used by Merritt [Ref. 14. p. 83-94] involves linearizing the critical-center spool valve flow equations and determining three valve coefficients: flow gain (K_q), flow pressure (K_c) and pressure sensitivity (K_p). Unfortunately this approach leads to difficulties if the

value position of interest lies at the origin ($K_{po} = \infty$, $K_{co} = 0$). To account for both the non-linear effects under large loading conditions and singularities introduced by linearization, an alternate approach retaining the orifice non-linearity, makes use of the following three equations:

$$Q_L = K_v \, i \, [P_s - P_L]^{\frac{1}{2}} \tag{5}$$

$$Q_L = D_m \, \dot{\theta}_m + C_{ip} \, P_L + C_{cp} \, \dot{P}_L \tag{6}$$

$$P_L \, D_m = J_t \, \ddot{\theta}_m + B_p \, \dot{\theta}_m + G_m \, \theta_m + T_L \tag{7}$$

Q_L = volume flow to the load [in^3/sec]

K_v = gain constant [$in^3/($ sec-ma-$\{psi\}^{\frac{1}{2}}$ $)$]

i = current [milliamps]

P_s = supply pressure [psi]

P_L = pressure drop across the load [psi]

D_m = volumetric displacement [in^3/rad]

θ_m = angular position [rad]

C_{ip} = combined leakage coefficient [(in^3-sec)/psi]

C_{cp} = compressibility coefficient [in^3/psi]

J_t = mass moment of inertia [lb-in-sec^2]

B_p = viscous damping coefficient [lb-in-sec]

G_m = torsional spring coefficient [in-lb/rad]

T_L = load torque [in-lb]

Equation (5) relates the non-linear effect of pressure and valve position to the load flow. Equation (6) is a flow

balance combining the effects of leakage, compressibility and actuator displacement to make up the load flow. Finally Eq. (7) expresses the dynamic equilibrium between the torque delivered and that required by the load. The block diagram in Fig. 9. represents the interaction of these equations. Further modifications of the non-linear model incorporating the effects of stiction and friction as well as saturation limits, are required by the nature of the physical plant and are incorporated into the simulation by means of MATRIXx super blocks, figures 10-22.

D. LOAD MODULE

The dynamometer utilizes magnetic saturation of a ferromagnetic core to provide a means of developing torque. The magnetic flux density (**B**) is related to the magnetic field intensity (**H**) and the magnetization (**M**) by the permeability (μo) as follows :

$$\mathbf{B} = \mu o \ (\mathbf{H} + \mathbf{M}) \qquad (8)$$

Ferromagnetic materials are anisotropic and the permeability is a tensor quantity which leads to non-linear behavior [Ref. 15. p. 310]. Additionally the torque developed by the dynamometer has been found to be a function of the rotational velocity of the input shaft.

System identification techniques were utilized to determine the module parameters from test data. The simplest characterization of the dynamometer load response indicated a time delay of 0.15 seconds and first order lag with a time constant of 2.0 seconds. The characterizing polynomial that converts command voltage to torque was found to be adequately represented by Eq. (9).

$$T_L = 0.03\ V^3 + 0.58\ V^2 - 0.71\ V + 0.01 \qquad (9)$$

where :

T_L = load torque [N-m]

V = voltage applied [VDC]

III. SIMULATION

Simulation of real systems first requires the definition of an appropriate mathematical model. As discussed previously these models are derived either from first principles, analysis of data that yields governing parameters, or both. For the system under investigation, this involves the analysis of continuous and discrete non-linear systems. This investigation adopts a modular approach to the simulation of the overall system.

A. DESCRIPTION

The simulation of the overall system (Fig. 10.) and its modules was undertaken using a commercially available software package (MATRIXx). In keeping with the goals of this investigation, all software support must function within the limitations of the selected microcomputer. The MATRIXx software module, SYSTEM_BUILD, allows the user to combine continuous, discrete and multiple rate modules into an integrated system simulation model. Inclusion of linear elements is accomplished with relative ease.

System simulation begins by on-screen construction of "simple" sub-system block diagrams selected either from a large collection of simple blocks (absolute value,

saturation, integration, etc.) or from user supplied routines. These collections of simple block diagrams, designated as super blocks, are then combined to represent more complicated systems. At each level, a combination of up to six blocks may be manipulated by the user. After construction of the block representation the user initiates an internal verification and analysis routine that checks simulation logic. If no errors are discovered the user selects an integration technique and runs the simulation. Changes to the simulation model are easily accomplished by modifying block parameters or swapping connections. The user need not code changes, other than user supplied routines, as is required with most main frame simulation languages. A complete listing of input and output variables is contained in Appendix D.

The results of several system simulations run on the microcomputer were compared to those using Digital Simulation Language (DSL) on an IBM 3033 main frame. With like integration techniques, the simulation output agreed to six significant digits.

B. CONTROL MODULE

Simulation of the control module (CTRL), presented in Fig. 11., is straightforward. Proportional or proportional plus integral (PI) control are employed by the control program software. The super blocks P_CTRL (Fig. 12.) and

T_CTRL (Fig. 13.) are discrete models in keeping with the nature of the digital system, and a sampling frequency is assigned during block definition. The module performs a sample and hold operation on both the reference (position and torque) and the feedback variables. The control error is calculated by a summing junction and converted to a quantized analog output by the D/A board. Limitations of both the programming language (size of error variable) and the I/O board (voltage output) are represented by saturation blocks. The inclusion of calculation times can be initiated either within the super block (time delay) or during simulation. The present control module logic allows I/O only at the end of each calculation so that the sampling interval and cycle period are synchronous.

C. ACTUATOR MODULE

Attempts to linearize the actuator module about its steady state, no load condition, gave rise to singularities in the governing equations. This translated into numerical ill-conditioning during simulation. Additionally, the phase plane trajectories (position, flow, etc.) are contained in an infinite set that is defined by previous states as well as control inputs.

Computer simulation allows the analysis of system behavior without the limitations imposed by the linearized model. Furthermore, if non-linearities (either hard or soft) dominate system response, a non-linear model is necessary in order to provide an accurate representation of the system.

Simulation of the actuator module (ACT), is represented by Fig. 14. The dynamic block, VI_AMP , represents the current amplifier. The gain block, TH_IND, converts the position (θ) to a voltage signal. The six main functional subdivisions of the actuator module are indicated in the figure and combined within the super block HPE (Fig. 15.). HPE is functionally equivalent to a hydraulic power element. Hard non-linearities include: load pressure and load flow saturation in super block AACT (Fig. 16.), absolute value of the load pressure drop in super block SACT (Fig. 17.), dead band (friction) in NACT (Fig. 18.), and a preload in XACT (Fig. 19.). The square root non-linearity, characteristic of valve controlled systems, is incorporated in the SACT super block. With this model the actuator will stall when subjected to an applied torque less than or equal to the magnitude of the stiction value.

D. LOAD MODULE

Simulation of the load module (LOAD) represented in Fig. 22, consists of a single variable characterizing block

that maps the input voltage to the output torque in accordance with Eq. (9). The time delay (DELAY) and first order pole (LAG) were derived from the data collected during system identification trials for the hysteresis dynomometer. Finally a gain block converts the load torque to an output voltage level.

IV. VERIFICATION

A. OPEN LOOP ALIGNMENT

1. Evaluation of Parameter Values

Determination of steady state gains such as the gain constant for the voltage to current amplifier and frequency break points were accomplished for the most part from direct physical measurement. Interface conversions (software and amplifier gains) were adjusted to allow for maximum resolution of the signal commensurate with I/O and sensor limitations. Least squares regression techniques were used to evaluate the coefficients for the characterizing polynomial (CHAR, Fig. 22.) used to model the torque response of the load module.

Where possible, the manufacture's specifications (frequency break points, linearity of the RVDT, etc.) were also validated by direct experiment. In some instances this was impracticable (actuator effective piston area, actuator inertia). The values for the viscous damping coefficient (B_p), hydraulic fluid bulk modulus (ß), torsional spring coefficient (G_m) and combined leakage coefficient (C_{ip}) were "educated" guesses. Appendix C contains a complete listing of applicable parameter values used.

2. Validation of Modules

Open loop alignment and validation of the simulation modules was accomplished in a iterative process. First, the physical system was subjected to excitation of known amplitudes and frequencies (step and sinusodial inputs) and the response recorded. Second, the results of the simulation trials were compared to the actual responses and the model parameters (those not otherwise verified) appropriately adjusted. The simulation module was then modified to reflect model changes and the trial repeated. During each iteration of the procedure the basic assumptions inherent in the model were evaluated.

The open loop validation of the control simulation proved the most successful. All parameters in the control model were well defined and accessible to verification. Differences between the simulation response and that of the control module were within the precision of the DT 2821 I/O board.

During calibration and alignment of the dynamometer excessive dead band (time delay) and relatively slow response were observed in achieving ordered torque loads. Figure 23. illustrates the response of the dynamometer to step changes in command voltage. Time delays averaged between 0.1 and 0.2 seconds. Significantly, residual magnetic fields in the dynamometer prevented achieving

desired torque values in the decreasing step tests but had no discernable effect for step increases. These characteristics make the load module unsuitable for its intended purpose. As a consequence of its unsuitability in a transient mode, efforts beyond characterization of the step response of the load module were not pursued and simulation of the load module was limited to constant torque values.

The actuator simulation provided substantially accurate phase and amplitude response at low frequencies. Figure 24. shows a comparison of the response between the simulation and test data for a sinusoidal input having an amplitude of 0.85 volts and a frequency of 0.84 Hz. The 'clipping' of the test data curve was attributed to the loss of supply pressure. As frequency increased the response of the actuator was observed to become increasingly non-linear. Figure 25. shows the open loop response for a range of frequencies while maintaining a constant signal amplitude well below that required to fully open the servovalve. The "clipped" portion of the waveforms represents stalling of the actuator. The angular velocity of the actuator, as indicated by the slope of the position trace, was essentially constant indicating a constant flow condition existed. The stall was increasingly evident as the signal frequency was raised. Simulation predicted a stall due to the stiction of the actuator but the onset of the observed

stall was much sooner than predicted. Subsequent analysis revealed that the dynamic characteristics of the hydraulic plant, characterized by supply pressure transients contribute to the stall. These effects, which dominate at frequencies above 1 Hz, were not accounted for in the simulation.

B. CLOSED LOOP TESTING

Integration of the simulation modules proceeded based on the known limitations of the individual module simulations. Closed loop testing was conducted by simulating the CYCLE program contained in Appendix B. The system responds to alternating reference positions of ±30 degrees. The period of the alternating input was determined by the time it took the system to achieve the reference position. Figure 26. compares the simulation and system response. The simulation response had almost the same frequency as the system response (4.5 vs 4.2 Hz.), but unlike the test data, did not exhibit constant angular velocity (indicative of constant flow). The simulation data shows evidence of proportional control action throughout its trace, as is indicated by the shape of the simulation response, but control action was observed in the system response only when the actual position approaches the reference position. Similar results were obtained for positions of ±10 and ±20 degrees.

Variation of simulation parameter values to compensate for the difference between test data and simulation results is not warranted by these results. To improve the simulation either the dynamics of the hydraulic power plant must be modelled or the hydraulic power plant modified to reduce the pressure transients.

V. CONCLUSIONS

The basic requirement that the microcomputer based system demonstrate the ability to control and direct the missile fin actuator was accomplished. The demonstration program PROFILE positions the actuator under varying loads and records the data for later analysis. The concept of modularizing the system to accommodate a wide variety of fin actuators remains to be demonstrated, but is strongly supported by the limited range of experience gained through this investigation. The currently available technological base provides equipment and software that are capable of implementing various control stategies, simulating complex non-linear and discontinuous systems and assisting the design engineer in the analysis of data.

VI. RECOMMENDATIONS

* The hydraulic power plant dynamics represent a serious limitation to achieving an acceptable model of the actuator. At a minimum, an accumulator should be added to the power plant to minimize the severity of the supply pressure transients and decouple the actuator and hydraulic plant dynamics.

* The pressure of the hydraulic fluid at various points in the system is central to a better understanding of the actuator model. The introduction of pressure sensors at the inlet and outlet of both the actuator and servovalve will allow better control and analysis of the actuator module.

* The dynamic load device was not appropriate for further investigation of the load response of candidate actuators. The Helac planetary hydraulic actuator, which in this investigation has served as the actuator, is a prime candidate for a load device. Not only would such a device have a much larger range of applied loads than the dynamometer, but the rotary actuator can apply a torque to a candidate actuator independent of its position or velocity.

* The analytical features of MATRIXx (SYSTEM_ID) combined with the power of the data acquisition system as represented by the DT 2821 I/O board and software, warrant a more detailed investigation of the actuator model. The actual values of the bulk fluid modulus (ß) and viscous damping coefficient (B_p) are excellent candidates for parameter identification.

* Reliance on analog components (hydraulic servovalves, etc.) tend to limit the performance of the system due to their inherently limited bandwidths. Recent advances in the development of digital components obviates the need for interface amplifiers and decreases the number of components. Procurement and investigation of an all-digital system is recommended.

* With additional sensors (pressure, temperature, etc.) the capacity exists for the development of a state space model. The obvious limitation of the microcomputer system is its limited amount of random access memory under the current operating system (640,000 bytes). However clever programming (including bank switching) may prove sufficient to handle the data arrays required. The use of direct memory access (DMA) has effectively decoupled the I/O and computational time, allowing manipulation of matrices during I/O cycles. An operational state space model will

allow the investigator to incorporate adaptive control algorithms into the control programs.

LIST OF REFERENCES

1. "Microsoft Pascal Compiler", MicroSoft Corporation, 1983.

2. Jensen, K. and Wirth, N., Pascal User Manual and Report. Second Edition. New York, Heidelberg, and Berlin: Springer-Verlag, 1980.

3. Beckwith, T. G., Buck, N. L., Marangoni, R. D., Mechanical Measurements, Reading, MA: Addison-Wesley Publishing Co., 1982.

4. "ATLAB Users Manual", Data Translation, Inc., 1986.

5. "Labtech Notebook", Laboratory Technologies Corporation, 1986.

6. Rosko, J. S., Digital Simulation of Physical Systems., Reading, MA: Addison-Wesley Publishing Co., 1972.

7. Åström, K. J., and Wittenmark, B., Computer Controlled Systems. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1984.

8. Borrie, J. A., Modern Control Systems. A Manual of Design Methods. London: Prentice-Hall, 1986.

9. Phillips, C. L., and Nagle, T. R. Jr., Digital Control System Analysis and Design. Englewood, NJ: Prentice-Hall, Inc., 1984.

10. Franklin, G. F., and Powell, J. D., Digital Control of Dynamic Systems. Reading, MA: Addison-Wesley, 1980.

11. Ogata, K., Discrete Time Control Systems. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1987.

12. Ogata, K., Modern Control Engineering. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1970.

13. Streeter, V. L., Fluid Mechanics. New York: McGraw-Hill, Inc., 1985.

14. Merritt, H. E., Hydraulic Control Systems. New York: John Wiley & Sons, 1967.

15. Hayt, W. H., Engineering Electro-Magnetics. New York: McGraw-Hill, Inc., 1974.

APPENDIX A

FIGURES

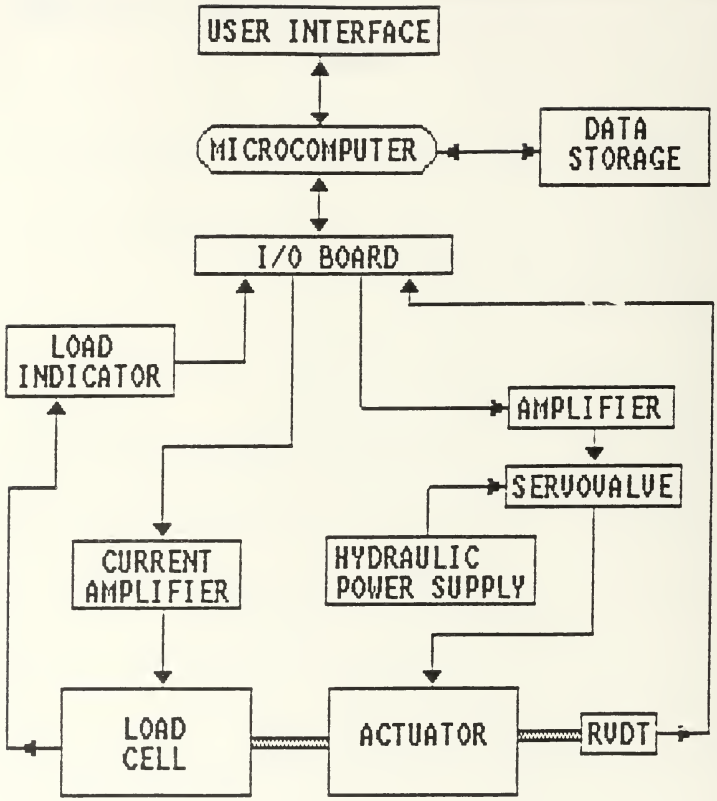


FIGURE 1. System Block Diagram

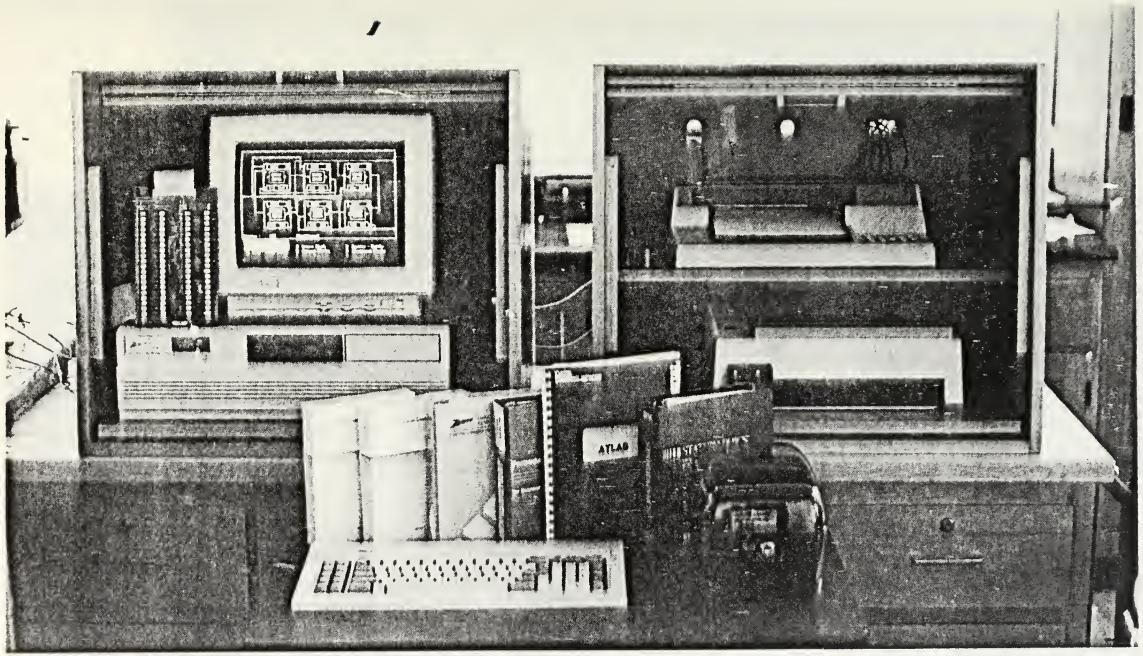


FIGURE 2. Control Module

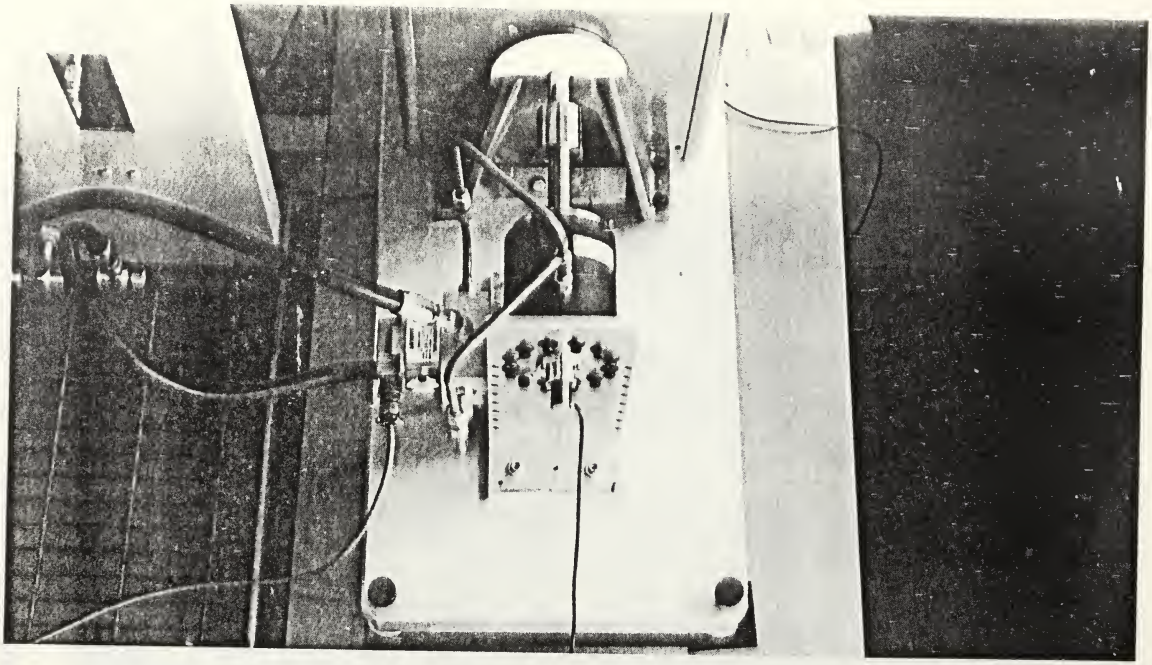


FIGURE 3. Actuator Module

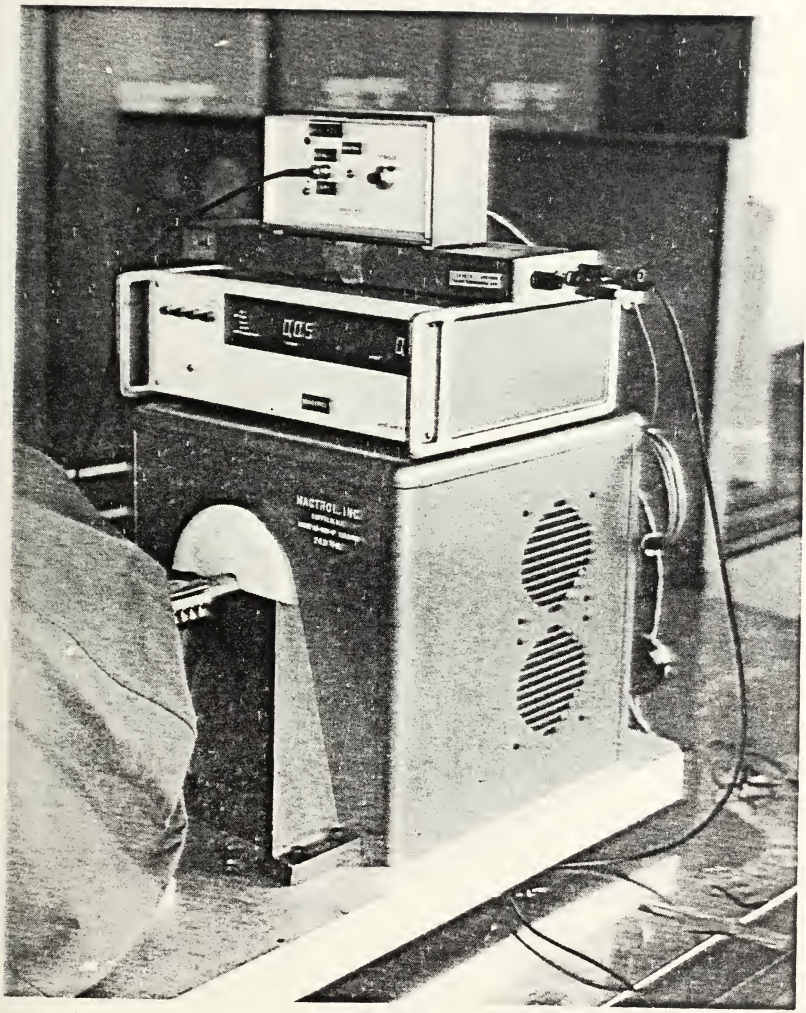


FIGURE 4. Load Module

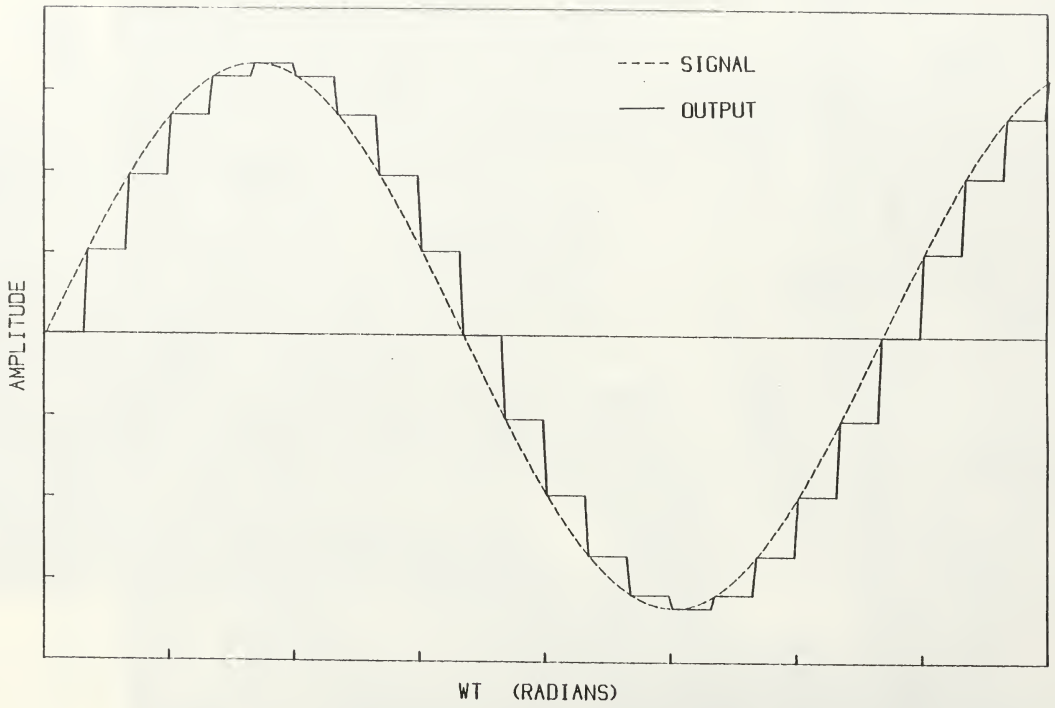


FIGURE 5. I/O BOARD SAMPLE AND HOLD

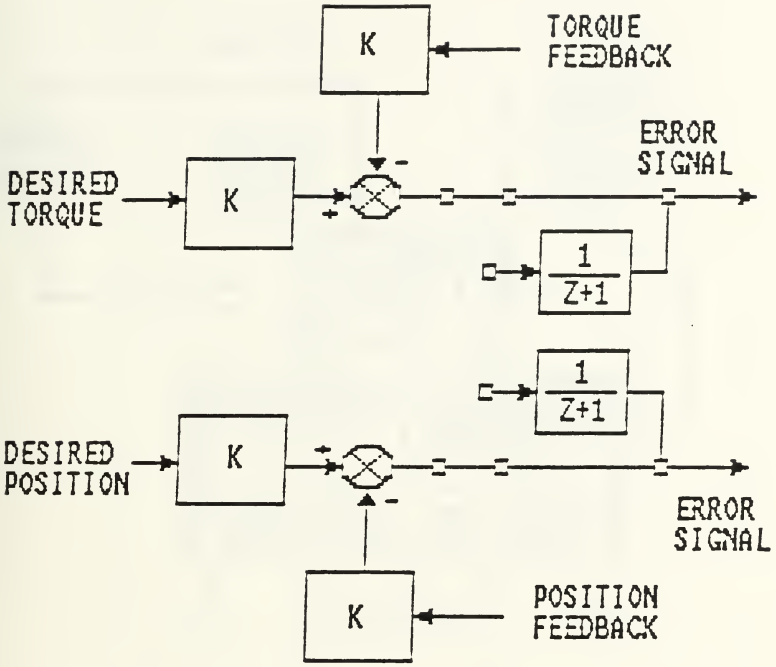


FIGURE 6. Control Module Discrete Block Diagram

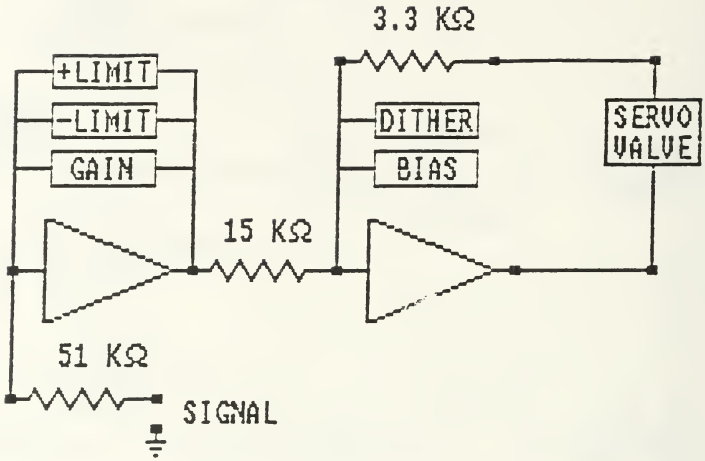


FIGURE 7. Current Amplifier Block Diagram

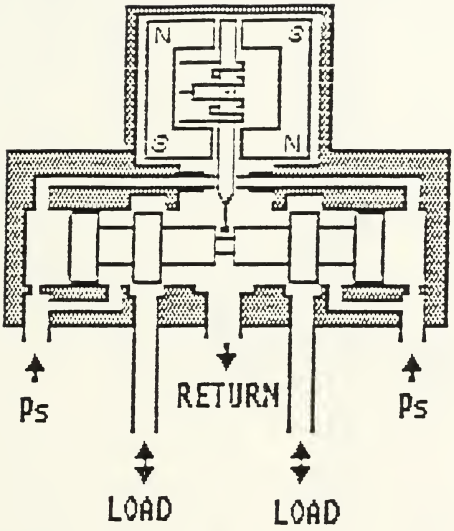


FIGURE 8. Electro-Hydraulic Servovalve

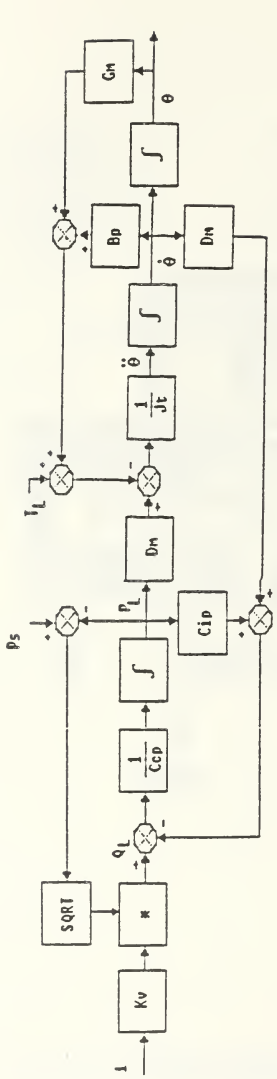


FIGURE 9. Simplified Actuator Block Diagram

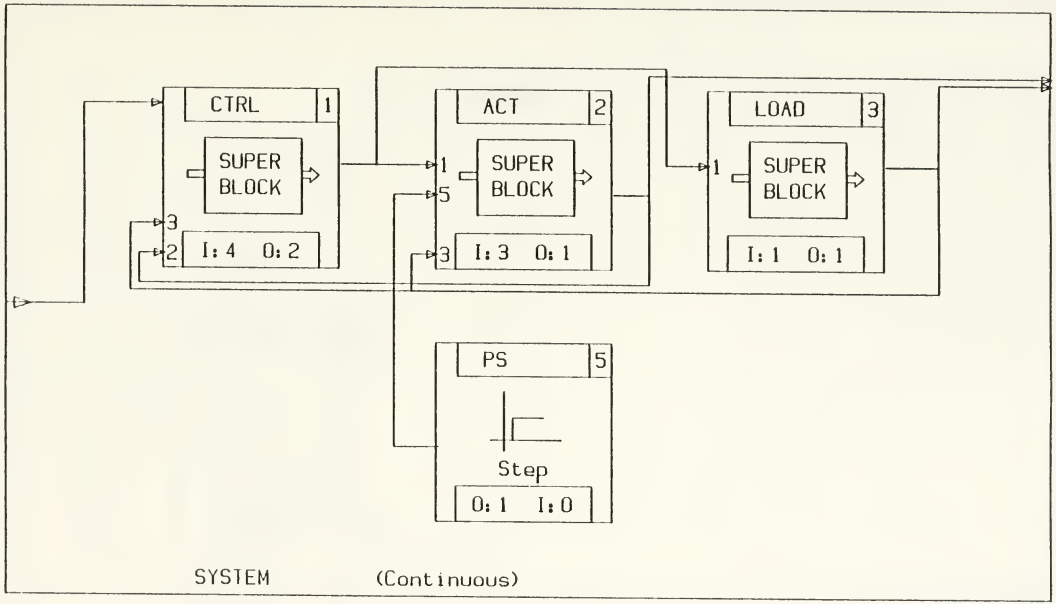


FIGURE 10. SYSTEM Simulation Block

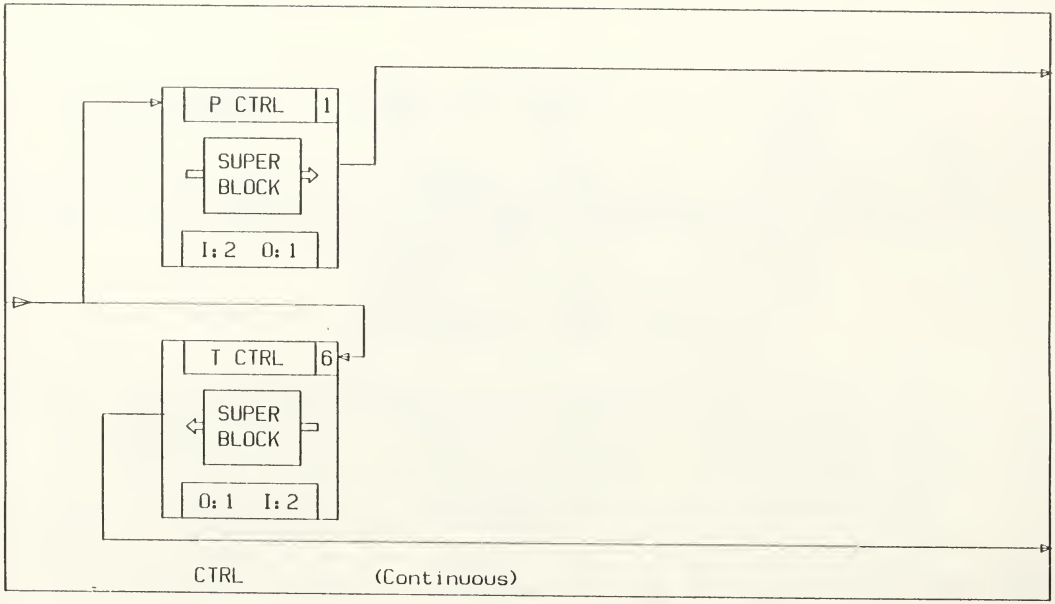


FIGURE 11. CTRL Simulation Block

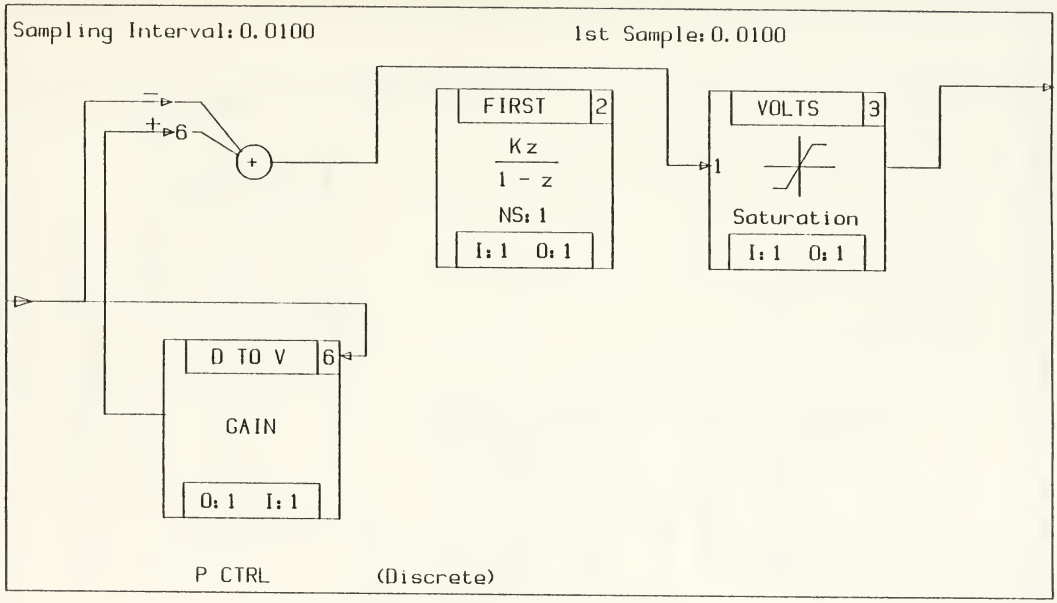


FIGURE 12. P_CTRL Simulation Block

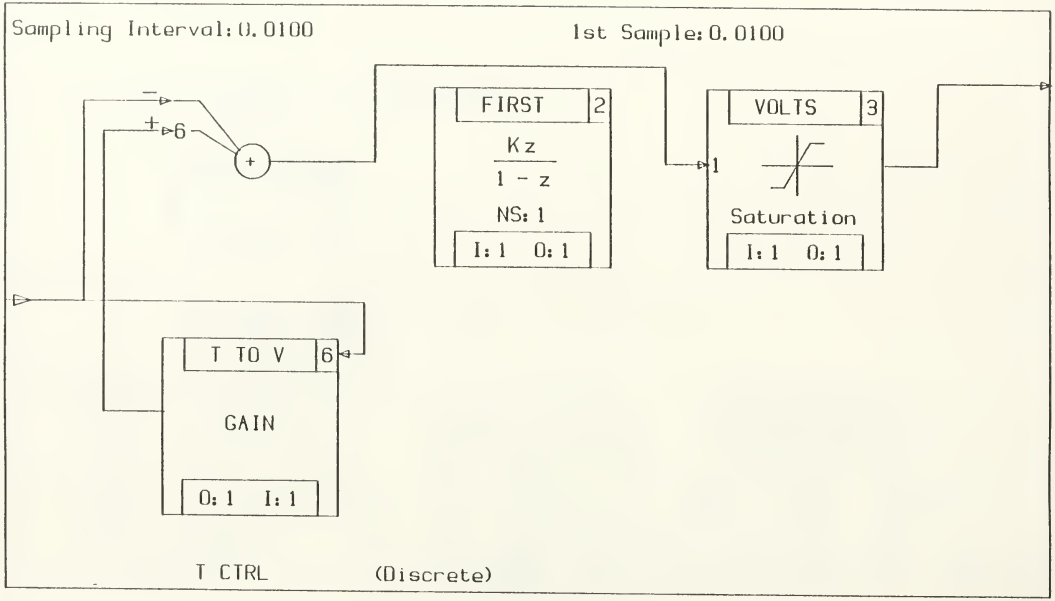


FIGURE 13. T_CTRL Simulation Block

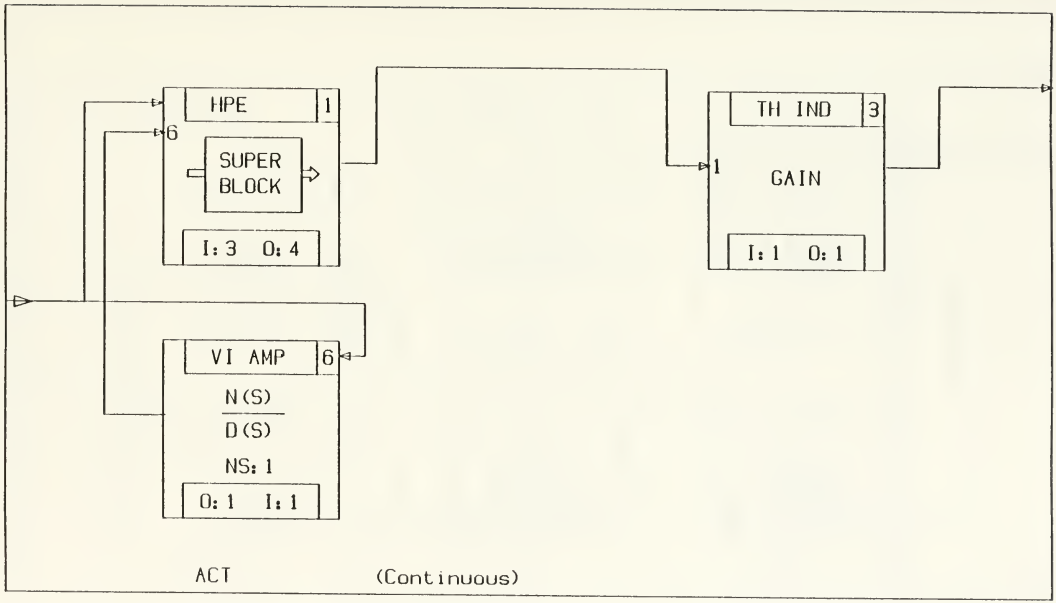


FIGURE 14. ACT Simulation Block

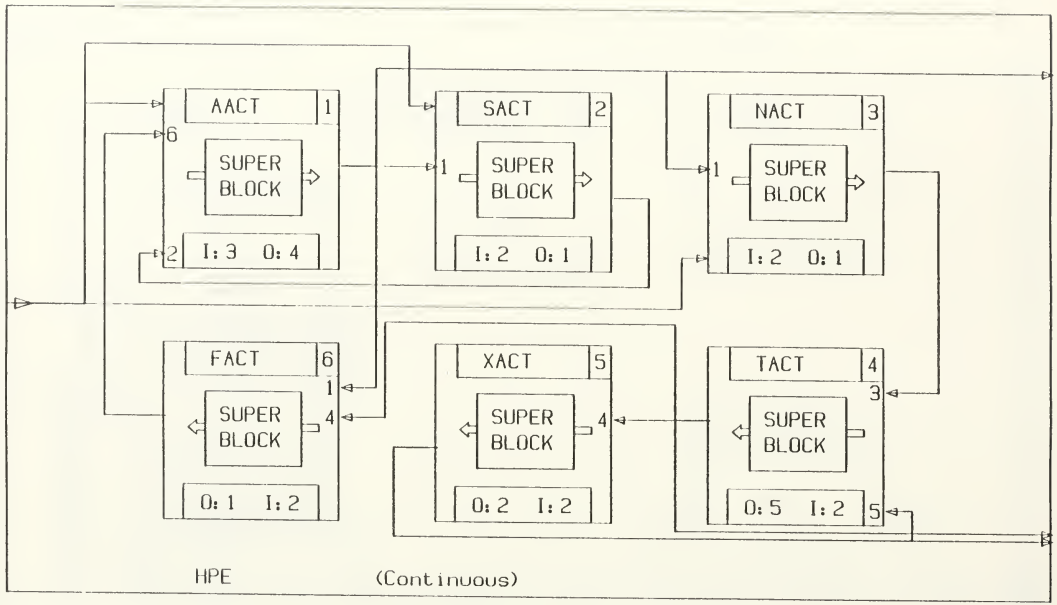


FIGURE 15. HPE Simulation Block

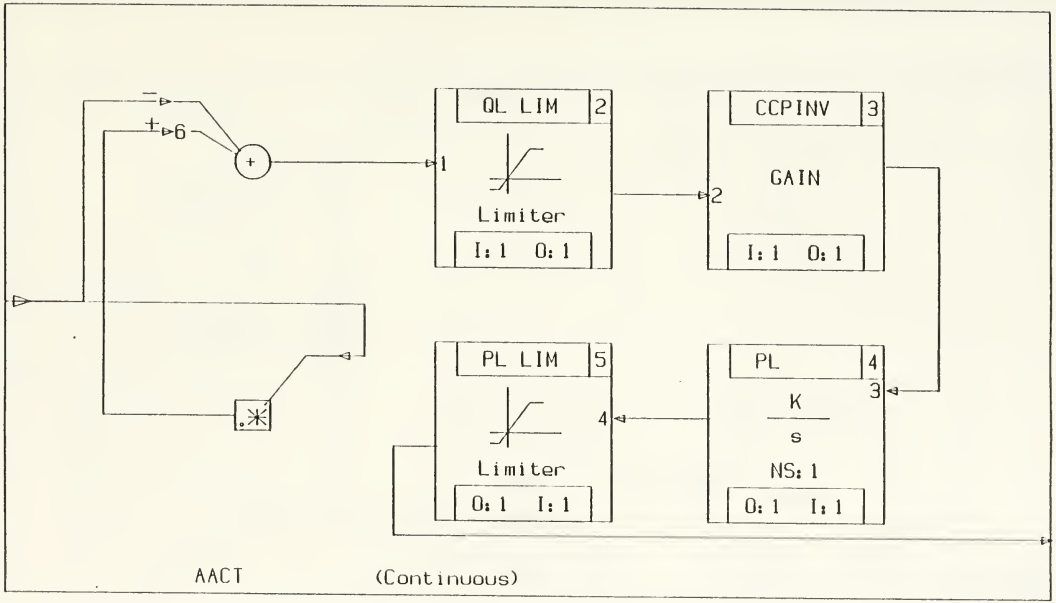


FIGURE 16. AACT Simulation Block

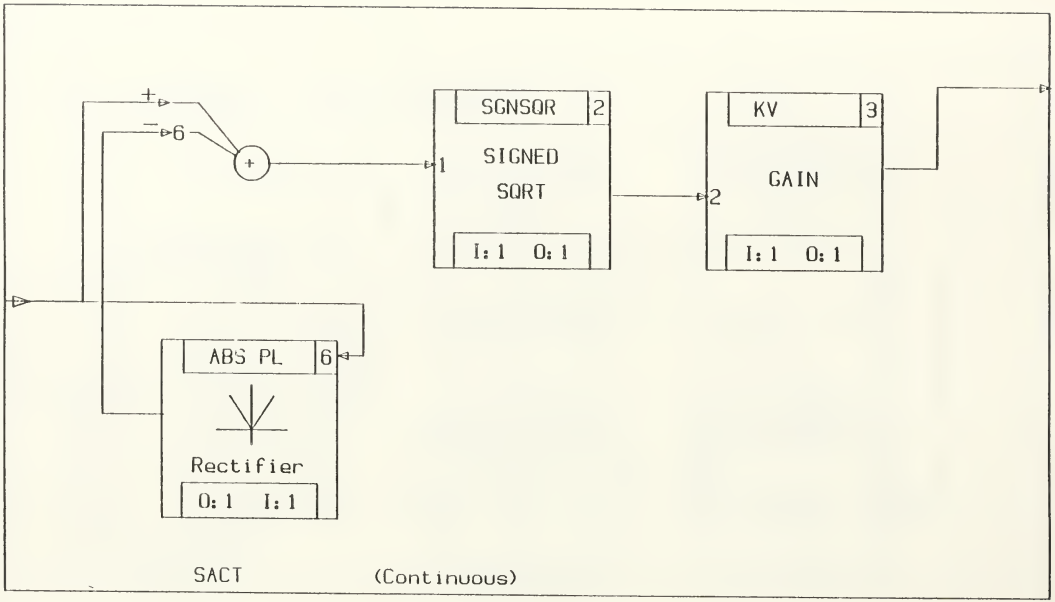


FIGURE 17. SACT Simulation Block

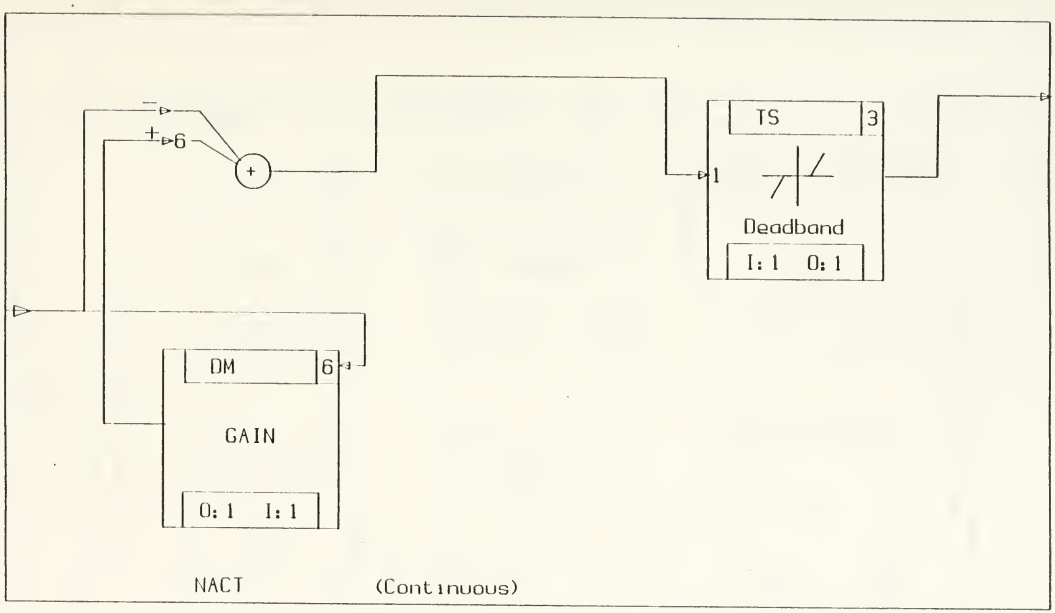


FIGURE 18. NACT Simulation Block

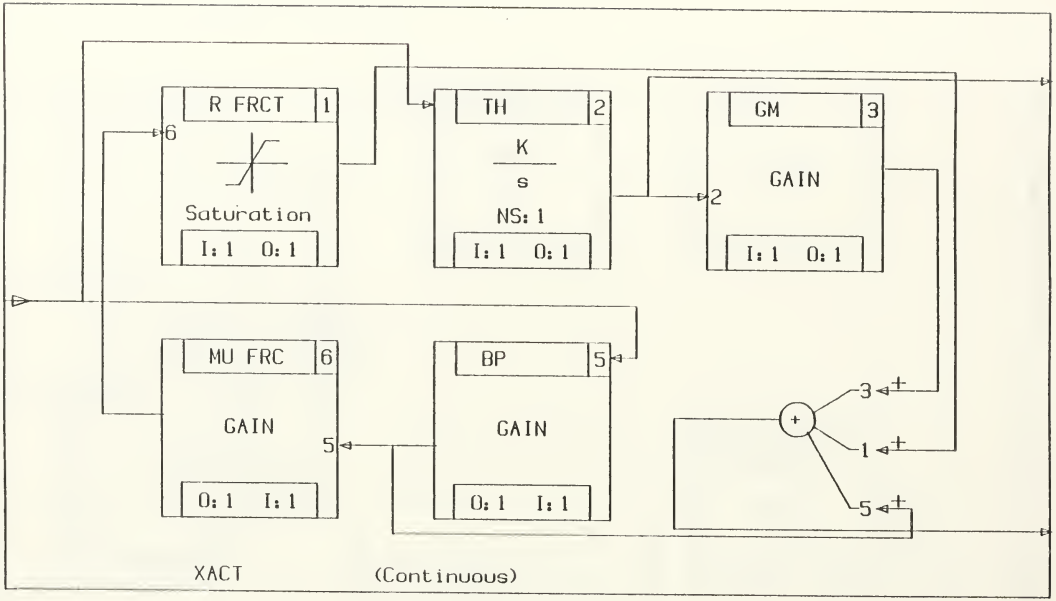


FIGURE 19. XACT Simulation Block

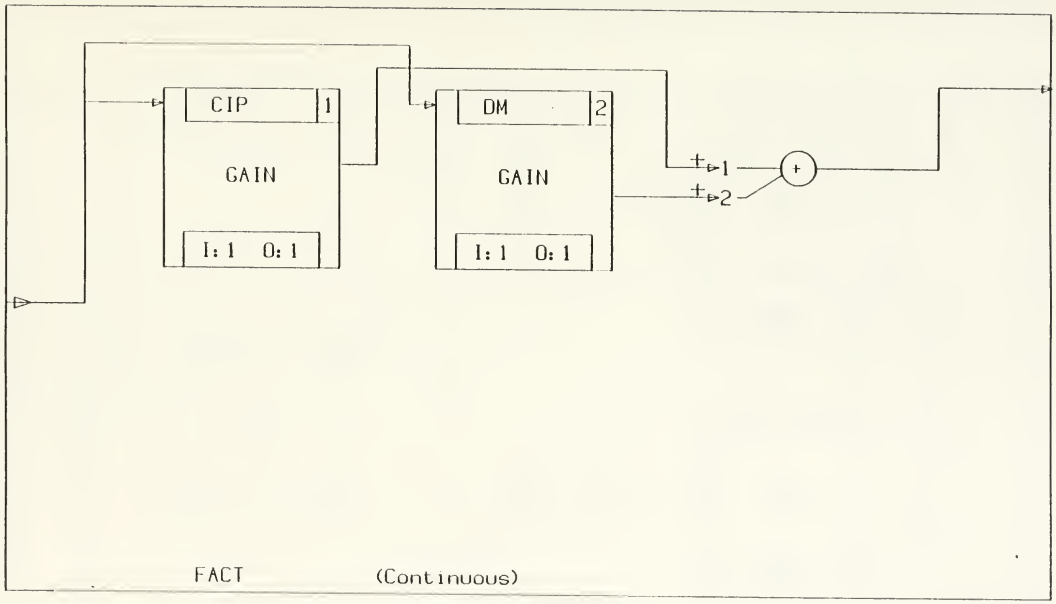


FIGURE 20. FACT Simulation Block

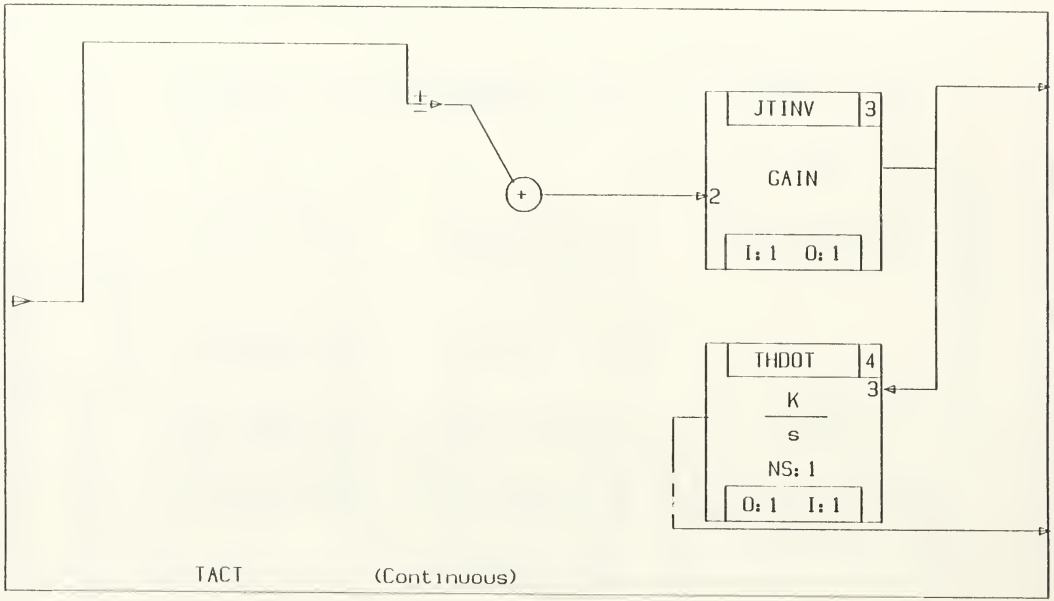


FIGURE 21. TACT Simulation Block

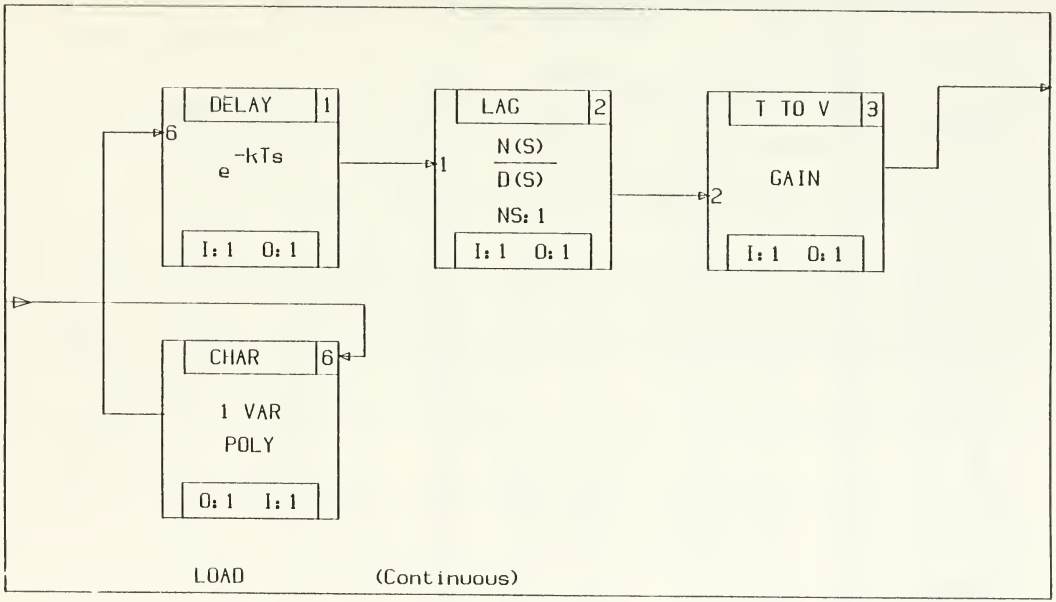


FIGURE 22. LOAD Simulation Block

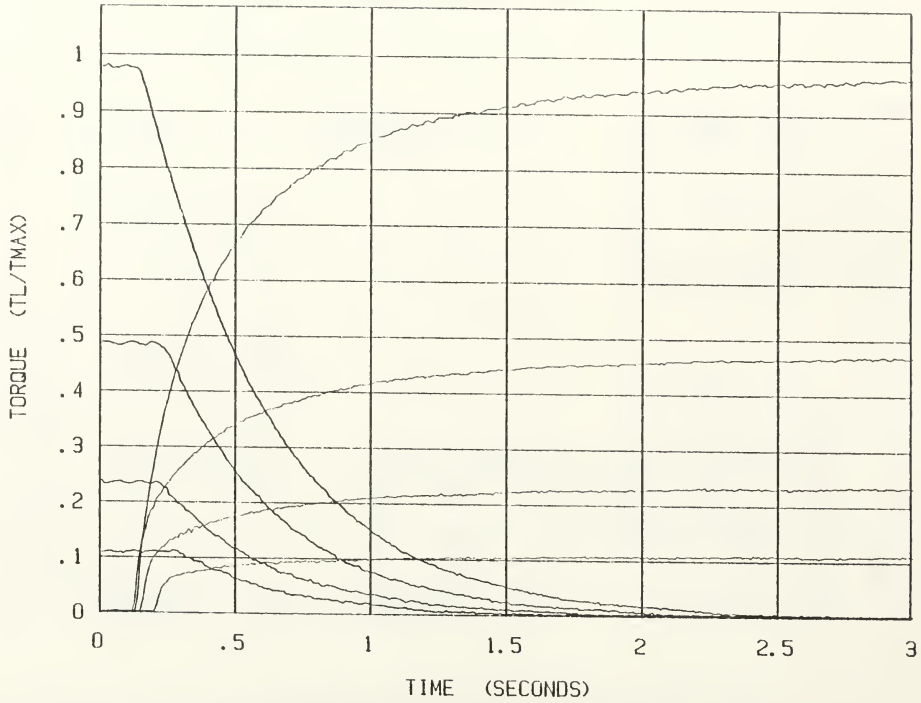


FIGURE 23. DYNAMOMETER STEP RESPONSE

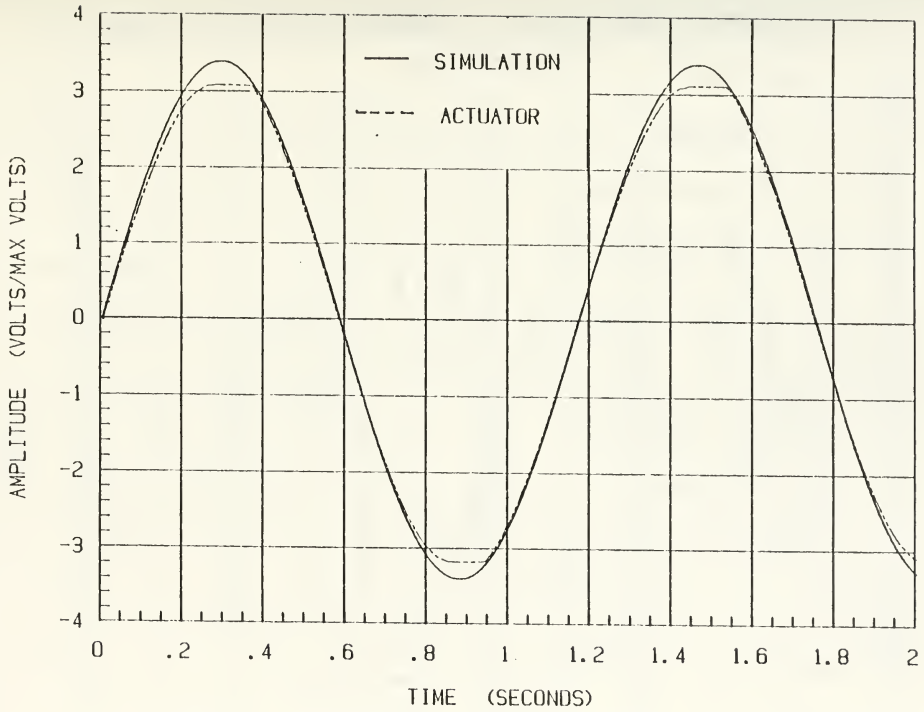


FIGURE 24. ACTUATOR VS SIMULATION (OPEN LOOP)

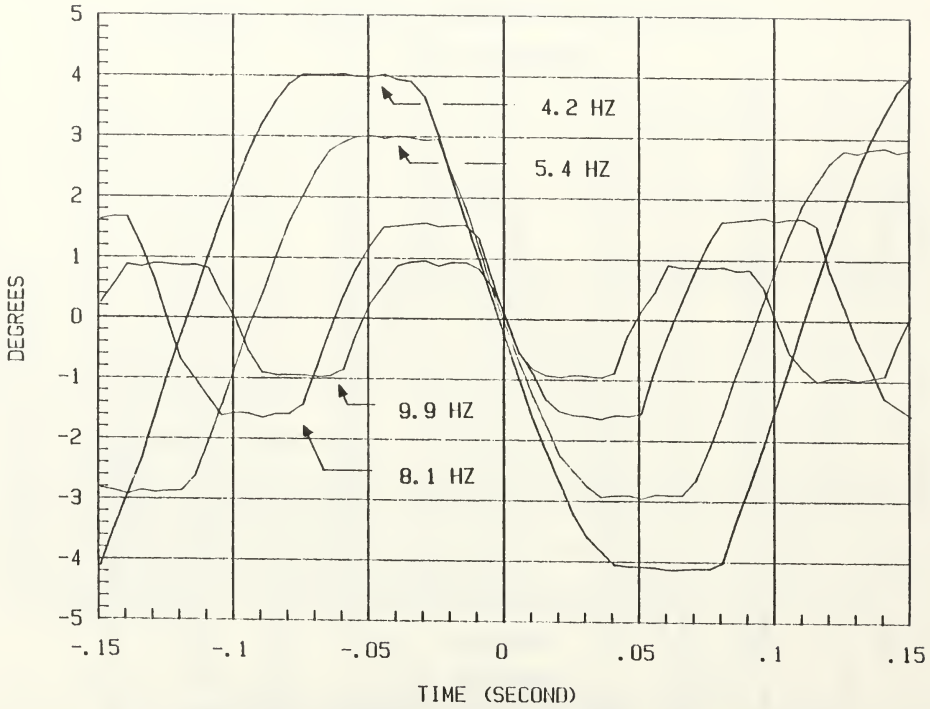


FIGURE 25. AMPLITUDE RESPONSE AT VARIOUS INPUT FREQUENCIES

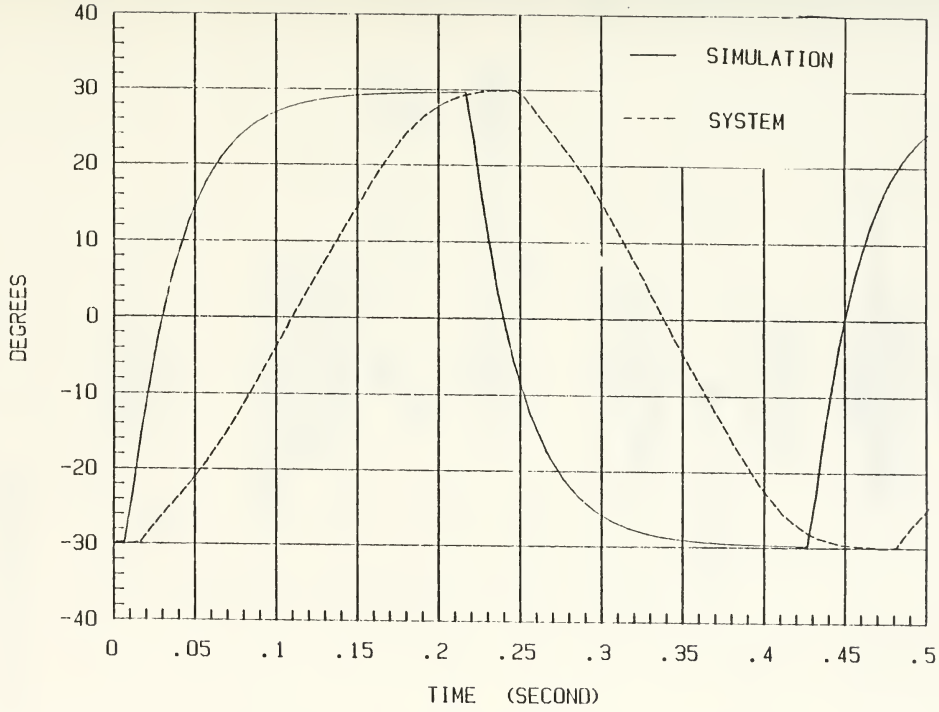


FIGURE 26. POSITION RESPONSE, SYSTEM VS SIMULATION

APPENDIX B

EXAMPLE CONTROL PROGRAMS

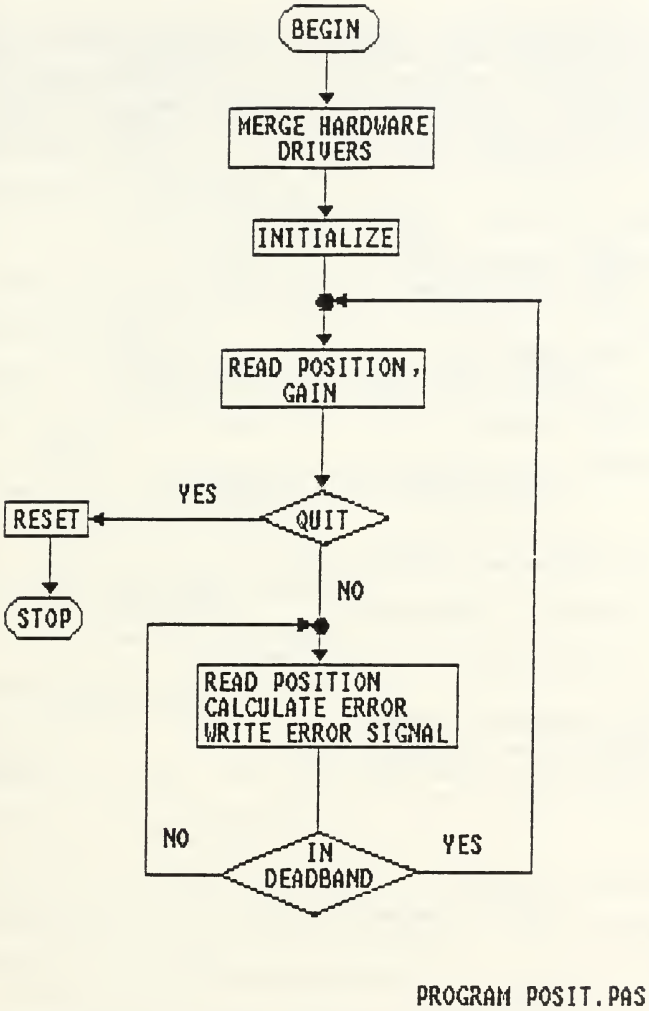
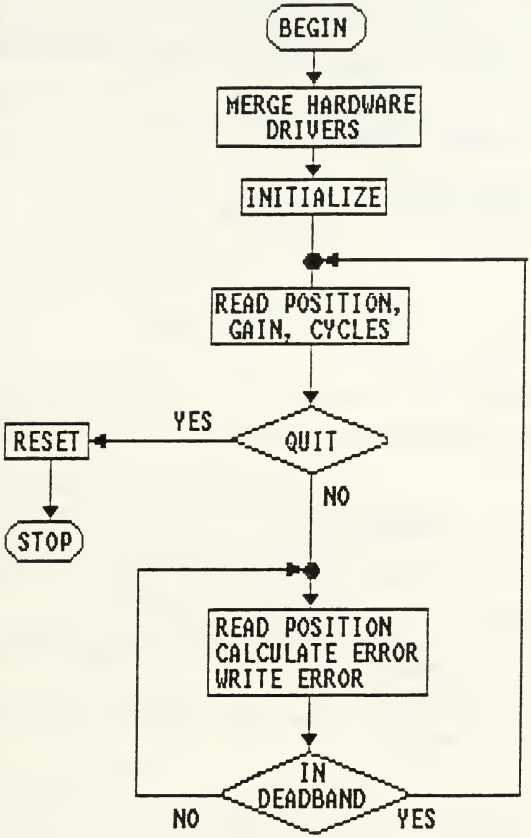


FIGURE 27. Program POSIT.PAS Flowchart

```
{ POSIT -- Position control of the HELAC rotary actuator
using RVDT position feedback.  This program is a simple
example of the implementation of negative feedback control.
The user may use either proportional control or proportional
+ integral control by enclosing the appropriate command
lines in brackets ({}). }


PROGRAM POSITION ( INPUT, OUTPUT );

{$include:'atldefs.pas'}
{$include:'atlerrs.pas'}

CONST
     dead_zone = 5;  { tolerance for error }

TYPE
     REL4 = real4;

PROCEDURE dump_configuration; EXTERN;

VAR

     ad_channels : CHANNEL_LIST;
                         { channel scan list }
     ad_gains : GAIN_LIST;
                         { gains for channels }
     configuration : AL_CONFIGURATION;
                         { storage for unit configuration
data }
     max_channel : INTEGER;
     data_value, feedback : WORD;
     status, pst, fdb, err, sig, aer, sign, i : INTEGER;
     posit, gain, temp : real4;



PROCEDURE CTRL_LOOP;
          { Simple feedback control loop with proportional
or proportional + integral control }
begin
          repeat     { begin position control loop }
          begin
               status := al_adc_value ( 1, 1, feedback );
{ get the current position from RVDT}
{ assumes a V+ max of 10 VDC }
               fdb := ord( feedback );
{ word to integer conversion }
               err := pst - fdb;
               if abs(err) > 4095 then err := 0;
```

```
{ summer of error }
                writeln('pos ',pst:5,'  fdb ',fdb:5);
{ display on screen }

                sig := trunc( gain * err );
{ proportional gain }
                {sig := sig + trunc( gain * err );}
{ proportional + integrater }

                if abs(sig) > 2048 then
                      sig := 2047 * (sig div abs(sig));
{ amplitude limit non-linearity }
                writeln('err ',err:5,'  sig ',sig:5);
{ display on screen }}
                data_value := wrd( sig + 02048 );
{ convert to word and scale zero for DT2821 board }
                status := al_dac_value ( 1, data_value );
{ output voltage to VICKERS ctrl amp TB1-a-4 }
           end;
           until ( abs( err ) < dead_zone );
{ continue loop until satisfied or interrupted }
           data_value := 02048;
           status := al_dac_value ( 1, data_value );
           sig := 0;
{ insure sig is zero }
end;    { Proc ctrl_loop }



BEGIN      { main program }
     writeln('POSITION -- User feedback position control
program.');
     writeln;

     status := al_initialize ;
{ initialize the ATLAB subroutines }

     status := al_select_board ( 1 );
{ address board 1, the first unit }

     status := al_reset ;
{ perform a reset on the device }

     dump_configuration ;
{ display the current unit configuration }

{ explain program }
     writeln;
     writeln('This program accepts a value in degrees and
positions the actuator.');
```

```
     writeln('Positive values are counter clock wise.
Output to the Vickers');
     writeln('control box is through DAC channel 1.');
     writeln;
{ define unit configuration for the number of A/D channels }
     status := al_get_configuration ( configuration );
     max_channel := configuration.channel_count - 1;
{ define channel and gain for DT2821 board. }
     ad_channels[0] := 1;  { select channel 1 }
     ad_gains[1] := 1;     { select gain = 1 }
     posit := 1.0;
     sig := 0;
     writeln;
     write('enter desired gain: ');
     readln( gain );
{ get gain  }
     if (gain <= 0.0) then gain := 1.0;
{ insure positive value }
     while ( posit <> 33.0 ) do
     begin
     writeln;
          write('enter desired position : ');
          readln( posit );
{ limit posit for present configuration }
          if (posit > 45.0) then posit := 1.0;
          if (posit < -45.0) then posit := -1.0;
{ integer value and scaling for 10 volts }
          pst := trunc( posit * 24.1 + 2048 );
{ 90 degrees full scale DT2821 board }
          err := 0;
{ call the control loop }
          ctrl_loop;
     end;
{ now set position to zero }
     pst := 02048;
     ctrl_loop;
     data_value := 2048;
     status := al_dac_value ( 1, data_value );
{ set voltage/posit to zero }
     writeln;
     writeln('  Program terminated normally ');
     status := al_terminate ;
{ terminate ATLAB operations }
END.
```



PROGRAM CYCLE.PAS

FIGURE 28. Program CYCLE.PAS Flowchart

```
{ CYCLE -- Position control of the HELAC rotary actuator
using RVDT position feedback.  This program uses the basic
position control block to exercise the actuator prior to
test runs.  Sufficient cycles should be run to insure even
temperature distribution throughout those components in
contact with hydraulic fluid.  }


PROGRAM CYCLES( INPUT, OUTPUT );

{$include:'atldefs.pas'}
{$include:'atlerrs.pas'}



CONST
     dead_zone = 4;  { tolerance for error }

TYPE
     REL4 = real4;

PROCEDURE dump_configuration; EXTERN;

VAR

     ad_channels : CHANNEL_LIST;
                         { channel scan list }
     ad_gains : GAIN_LIST;
                         { gains for channels }
     configuration : AL_CONFIGURATION;
                  { storage for unit configuration data }
     max_channel : INTEGER;
     data_value, feedback : WORD;
     status, pst, fdb, err, sig, aer, sign, i, n : INTEGER;
     posit, gain : real4;



PROCEDURE CTRL_LOOP;
begin
          repeat     { begin position control loop }
          begin
               status := al_adc_value ( 1, 1, feedback );
{ get the current position }
{ assumes a V+ max of 10 VDC }
               fdb := ord( feedback );
{ word to integer conversion }
               err := pst - fdb;
{ summer of error }
               sig := trunc( gain * err );
{ proportional gain }
```

```
                {sig := sig + trunc( gain * err );}
{ proportional + integrater }
                if abs(sig) > 2048 then
                      sig := 2047 * (sig div abs(sig));
{ amplitude limit non-linearity }
                data_value := wrd( sig + 02048 );
{ convert to word and scale zero for DT2821 board }
                status := al_dac_value ( 1, data_value );
{ output voltage to VICKERS ctrl amp TB1-a-4 }
           end;
           until ( abs( err ) < dead_zone );
           data_value := 02048;
           status := al_dac_value ( 1, data_value );
           sig := 0;
end;    { Proc ctrl_loop }



BEGIN
     writeln('CYCLE -- User defined exercise control
program.');
     writeln;

     status := al_initialize ;
{ initialize the ATLAB subroutines }

     status := al_select_board ( 1 );
{ address board 1, the first unit }

     status := al_reset ;
{ perform a reset on the device }

{    dump_configuration ;}
{ display the current unit configuration }

{ explain program }
     writeln;
     writeln(' This program accepts a value in degrees and
cycles the actuator N times.');
     write(' Enter N : ');  readln(n);  writeln;
     write(' Output to the Vickers');
     writeln('control box is through DAC channel 1.');
     writeln;

{ define unit configuration for the number of A/D channels }
     status := al_get_configuration ( configuration );
     max_channel := configuration.channel_count - 1;

{ define channel and gain for DT2821 board. }
     ad_channels[0] := 1;  { select channel 1 }
     ad_gains[1] := 1;     { select gain = 1 }
```

```
     posit := 1.0;
     sig := 0;

     while ( posit <> 0.0 ) do
     begin
          writeln;
          write('enter desired gain: ');
          readln( gain );
          if (gain <= 0.0) then gain := 1.0;
          write('enter the position : ');
          readln( posit );
          if abs( posit ) > 45.0 then posit := 45.0;
          writeln;
          for i := 1 to n  do
          begin
               posit := -posit;
               pst := trunc( posit * 24.2 + 2048 );
               err := 0;
               ctrl_loop;
          end;
     end;       { while }

END.
```

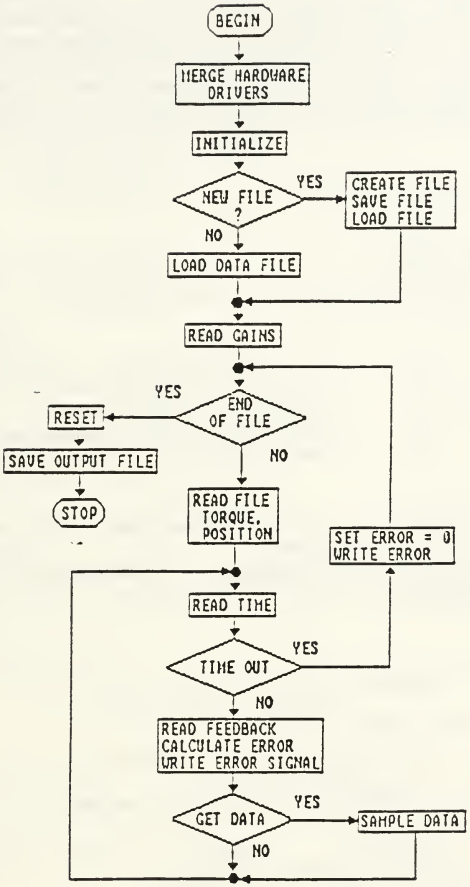


FIGURE 29. Program PROFILE.PAS Flowchart

```
{ PROFILE -- Basic demonstration program.  This program will
generate a position and torque profile for the Helac rotary
actuator, execute that profile and store data from the trial
for later analysis.  The user can adjust gains during
program execution but must modify the program to change the
minimum time interval for desired torques and fin positions.
}


PROGRAM PROFILE(input, output);

{$include:'atldefs.pas'}
{$include:'atlerrs.pas'}

FUNCTION tics : word; extern;

CONST
     deadband  =    5;   { a dead band tolerance    }
     ary_len   = 2500;   { length of data array     }
     time_len  =   20;   { duration of flight (sec) }
     n         =    2;   { num of div in 1 sec      }

TYPE
     ary_dat   = array [1..2,1..(n*time_len)] of integer;
     dat_file  = file of ary_dat;
     ary_dmp   = array [1..ary_len,1..5] of integer;
     dmp_file  = text;

VAR

     ad_channels : CHANNEL_LIST;
                        { channel scan list }
     ad_gains : GAIN_LIST;
                        { gains for channels }
     configuration : AL_CONFIGURATION;
                    { storage for unit configuration data }

     max_channel : INTEGER;
     data_value, pfdbck, tfdbck : WORD;
     status, pst, pfdb, perr, psig, paer,
     trq, tfdb, terr, tsig, taer, sign,
     i, j, k, int : INTEGER;
     ch : char;
     pgain, tgain : real4;
     flight : ary_dat;
     datadmp : array [1..5,1..ary_len] of integer;
     file_name : lstring(15);
     longstr :   lstring(80);
     F1 : dat_file;
     F2 : dmp_file;
```

```
PROCEDURE ZERO;
          { returns all outputs on DT 2821 to zero}
begin
     data_value := wrd(02048);
     status := al_dac_value( 0, data_value );
     status := al_dac_value( 1, data_value );
{ set all outputs to zero }
end;     { Proc Zero }



PROCEDURE SAVE_DATA;
          { stores data in a file on the virtual disk }
begin
     page;
     writeln(' Storing data in G:flight.dat');
     writeln;

     file_name := 'G:flight.dat';
     assign(F2, file_name);
     rewrite( F2 );

     for i := 1 to ary_len do
     begin
          writeln(F2, datadmp[1,i]:7, datadmp[2,i]:7,
datadmp[3,i]:7, datadmp[4,i]:7, datadmp[5,i]:7 );
     end;

     close( F2 );
end;     { Proc Save_data }



PROCEDURE BUILD;
          { Generates a profile based on user input }
var
     time, otime : integer;
     post, torq : real4;
begin
     otime := 0;
     page;
     repeat
     writeln;
     writeln('flight ends at time = ',time_len:3,' sec');
     writeln;
     write('Enter the end time of the event (sec) : ');
     readln(time);  time := 2 * time;
     if time > otime then
     begin
          write('Enter the desired position (deg) : ');
```

```
          readln(post);
          if post >  45.0 then post :=  45.0;
          if post < -45.0 then post := -45.0;
{ integer value and scaling for 10 volts }
          pst := trunc(post * 24.1 + 2048);
          write('Enter desired torque (N-m) : ');
          readln(torq);
          torq := abs(torq);
          if torq > 24.0 then torq := 24.0;
{ integer value and scaling for 10 volts }
          trq := trunc(torq * 85.333 + 2048);

          for i := otime+1 to time do
          begin
               flight[1,i] := pst;  flight[2,i] := trq
          end;
          otime := time
     end;
     until (otime = (2*time_len));

     page;
     write('press space then return .. ');  readln(ch);
end;    { Proc Build }


PROCEDURE INIT;
          { initializes the DT 2821 and the data arrays}
begin
     status := al_initialize ;
{ initialize the ATLAB subroutines }

     status := al_select_board ( 1 );
{ address board 1, the first unit }

     status := al_reset ;
{ perform a reset on the device }

     Zero;   { zeros output of I/O board }

     writeln;
     ch := ' ';
     write(' Use an existing Flight Profile (Y/N) ? ');
     readln(ch);
     if (ch = 'Y') or (ch = 'y') then
     begin
          writeln;
          write(' Enter file name : ');
          readln( file_name );
          assign( F1, file_name );
          reset( F1 );
```

```
          get( F1 );
          flight := F1^;       { retrieve data from file }
          close( F1 );
     end
     else
     begin
          file_name := 'temp.dat';
          assign ( F1, file_name);
          rewrite( F1 );
          Build;
          F1^ := flight;
          put( F1 );           { save data to file      }
          close( F1 );
     end;

     psig := 0;  tsig := 0;  perr := 0;  terr := 0;
     k := 1;
     page;
     writeln(' Select analog input on magtrol torque
control');
     writeln;  write(' press space then return .');
     readln(ch);

     writeln;
     write('enter desired position gain: ');
     readln( pgain );
     if (pgain <= 0.0) then pgain := 1.0;
     write('enter desired torque gain: ');
     readln( tgain );
     if (tgain <= 0.0) then tgain := 1.0;

end;       { end Init }


PROCEDURE POSIT;
          { basic control loop for position}
begin
          begin
               status := al_adc_value ( 1, 1, pfdbck );
{ get the current position }
{ assumes a V+ max of 10 VDC }
               pfdb := ord( pfdbck );
{ word to integer conversion }
               perr := pst - pfdb;
{ summer }
               psig := trunc( pgain * perr );
{ proportional gain }
               {psig := psig + trunc( pgain * perr );}
{ proportional + integrater }
               if abs(psig) > 2048 then
```

```
                    psig := 2040 * (psig div abs(psig));
{ amplitude limit non-linearity }
                data_value := wrd( psig + 02048 );
{ convert to word and scale zero for DT2821 board }
                status := al_dac_value ( 1, data_value );
{ output voltage to VICKERS ctrl amp TB1-a-4 }
          end;
end;   { Proc POSIT }



PROCEDURE TORQUE;
          { basic control loop for torque}
begin
          begin
                status := al_adc_value ( 0, 1, tfdbck );
{ get the current torque value                    }
{ assumes a 41.66/1 VDC amp from MAGTROL 4618 }
                tfdb := ord( tfdbck );
{ convert to integer value }
                tfdb := abs( 2048 - tfdb) + 2048;
{ positive value independent of direction of rotation }
                terr := trq - tfdb;
{ summer }
                taer := abs( terr );
                if (taer>0) then sign := trunc(terr/taer)
                else sign := 1;
                tsig := trunc( tgain * terr );
{ proportional gain }
                {psig := psig + trunc( tgain * terr );}
{ proportional + integrater }
                if ((tsig < 0) or (abs(perr) < deadband))
then tsig := 0;
{ prevents 'cogging' of rotor }
                data_value := wrd( tsig + 02048 );
{ convert to word and scale zero for DT2821 board }
                status := al_dac_value ( 0, data_value );
{ output voltage to MAGTROL 4637 }
          end;
end;   { Proc TORQUE }



PROCEDURE CTRL;
          { Controls timing of events and the data
acquisition }
begin
     i := i + 1;
     pst := flight[1,i];  trq := flight[2,i];
     while tics < 50 do
     begin
```

```
          Posit;
          Torque;
          if k <= ary_len then
          begin
               datadmp[1,k] := ord(tics);
               datadmp[2,k] := pst;  datadmp[3,k] := pfdb;
               datadmp[4,k] := trq;  datadmp[5,k] := tfdb;
               k := k + 1;
for j := 1 to 2000 do begin  int := 1 end;
          end;
     end;
     i := i + 1;
     pst := flight[1,i];  trq := flight[2,i];
     while tics >= 50 do
     begin
          Posit;
          Torque;
          if k <= ary_len then
          begin
               datadmp[1,k] := ord(tics);
               datadmp[2,k] := pst;  datadmp[3,k] := pfdb;
               datadmp[4,k] := trq;  datadmp[5,k] := tfdb;
               k := k + 1;
for j := 1 to 2000 do begin  int := 1 end;
          end;
     end;

end;    { Proc Ctrl }



BEGIN     { main program }

     Init;          { initializes board etc }
     page(output);  writeln('  Missile away .... ');
     i := 1;
     while i < (n*time_len-1) do
     begin
          Ctrl;
     end;
     writeln;  writeln('  Intruder destroyed ! ');
     Zero;    { zeros output of I/O board }
     status := al_terminate ; { terminate ATLAB operations }
     Save_data;    { saves data in flight.dat }
     writeln(' Select manual on magtrol torque control');
     writeln;
     write(' Press space then return ... ');  readln(ch);
     page;
     writeln('Program terminated normally.  Goode bye...');
END.
```

APPENDIX C

MODEL AND SIMULATION PARAMETERS

α = angular acceleration [rad / sec^2]

β = fluid bulk modulus [50,000 lb/in^2]

B_p = viscous damping coefficient [23.4 lb-in-sec]

C_{cp} = compressibility coefficient [5.58E-5 in^3/psi]

C_{ip} = leakage coefficient [8.1E-3 (in^3-sec)/psi]

D TO V = degrees to volts conversion [3.0 deg/volt]

D_m = rotational displacement [2.449 in^3/rad]

G_m = torsional spring coefficient [0.0 in-lb/rad]

J_t = mass moment of inertia [2.56E-2 lb-in-sec^2]

K_v = gain constant [.00352 in^3/(sec-ma-$(psi)^{\frac{1}{2}}$)]

P_s = hydraulic supply pressure [1000 psia]

θ = angular position [radians or degrees]

T_s = torque due to stiction [180 in-lb]

T TO V = torque to volts conversion [.4167 Volts/N-m]

TH IND = radians to voltage convertion [8.748 rad/volt]

V_T = total fluid volume [11.16 in^3]

ω = angular velocity [rad / sec]

APPENDIX D

SIMULATION I/O MAPS

BLOCK : SYSTEM

INPUT	OUTPUT
1. θ [degrees] (desired)	1. θ [degrees] (actual)
2. T_L [N-m] (desired)	2. T_L [N-m] (actual)

BLOCK : CTRL

INPUT	OUTPUT
1. θ [degrees]	1. θ [VDC] (error)
2. θ [VDC] (feedback)	2. T_L [VDC] (error)
3. T_L [N-m]	
4. T_L [VDC] (feedback)	

BLOCK : P_CTRL

INPUT	OUTPUT
1. θ [degrees]	1. θ [VDC] (error)
2. θ [VDC] (feedback)	

BLOCK : T_CTRL

INPUT	OUTPUT
1. T_L [N-m]	1. T_L [VDC] (error)
2. T_L [VDC] (feedback)	

BLOCK : ACT

INPUT	OUTPUT
1. θ [VDC) (error)	1. θ [VDC/rad]
2. P_s [PSIA]	
3. T_L [in-lb]	

BLOCK : HPE

INPUT	OUTPUT
1. i [ma]	1. θ [rad]
2. P_S [PSIA]	2. ω [rad/s]
3. T_L [in-lb]	3. α [rad/s^2]
	4. P_L [PSIA]

BLOCK : AACT

INPUT	OUTPUT
1. i [ma]	1-5. P_L [PSI]
2. [in^3/s-ma]	
3. Q [in^3/s]	

BLOCK : SACT

INPUT	OUTPUT
1. P_L [PSI]	1. [in^3/ma-s]

BLOCK : NACT

INPUT	OUTPUT
1. P_L [PSI]	1. T [in-lb]
2. T_L [in-lb]	

BLOCK : TACT

INPUT	OUTPUT
1. T [in-lb]	1-4. ω [rad/s]
2. T [in-lb]	5. α [rad/s^2]

BLOCK : FACT

INPUT	OUTPUT
1. P_L [PSI]	1. Q [in^3/s]
2. ω [rad/s]	

BLOCK : XACT

INPUT	OUTPUT
1-2. ω [rad/s]	1. θ [rad] 2. T [in-lb]

BLOCK : LOAD

INPUT	OUTPUT
1. T_L [VDC]	1. T_L [VDC] (feedback)

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Department Chairman, Code 69 Department of Mechanical Engineering Naval Postgraduate School Monterey, CA 93943-5000	2
4.	Professor R. H. Nunn, Code 69Nn Naval Postgraduate School Monterey, CA 93943-5000	5
5.	Professor D. L. Smith, Code 69Sm Naval Postgraduate School Monterey, CA 93943-5000	1
6.	Professor L. W. Chang, Code 69Ck Naval Postgraduate School Monterey, CA 93943-5000	1
7.	Professor A. Gerba, Jr., Code 62Gz Naval Postgraduate School Monterey, CA 93943-5000	1
8.	Mr. R. Geres, Code 32731 Naval Weapons Center China Lake, CA 93555	2
9.	Mr. R. Dettling, Code 3275 Naval Weapons Center China Lake, CA 93555	1
10.	Mr. A. Danielson, Code 3272 Naval Weapons Center China Lake, CA 93555	1
11.	LT G. L. Goode, USN Mare Island Naval Shipyard Vallejo, CA 94592-5100	3

322 - 898

Thesis
G557
c.1
Goode
Feasibility study of a microprocessor controlled actuator test mechanism.

Thesis
G557
c.1
Goode
Feasibility study of a microprocessor controlled actuator test mechanism.

AMERICAN BINDERY
TOPEKA, KANSAS

thesG557
Feasibility study of a microprocessor co
3 2768 000 78239 5
DUDLEY KNOX LIBRARY